ACL 2022

The 60th Annual Meeting of the Association for Computational Linguistics

Proceedings of the Conference, Vol. 2 (Short Papers)

May 22-27, 2022

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

Diamond

amazon | science

Bloomberg
Engineering

Google Research

LIVEPERSON

Meta

Platinum

Bai du 百度

ByteDance

DeepMind

grammarly

GTCOM

IBM

Megagon Labs

Microsoft

Gold

Alibaba Group
阿里巴巴集团

BOSCH
Invented for life

co:here

RESEARCH

Relativity

servicenow

Silver

ASAPP

duolingo

NAVER

Bronze

Adobe

Babelscape

Spotify

Order copies of this and other ACL proceedings from:

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

ISBN 978-1-955917-22-3

Message from the General Chair

Welcome to ACL 2022, the 60th Annual Meeting of the Association for Computational Linguistics! The conference will be held in Dublin, the capital of Ireland, on May 22–27, 2022.

ACL 2022 will be a hybrid conference. After two fully virtual editions, ACL 2020 and ACL 2021, due to the covid-19 pandemic, this year we are gradually coming back to normality, estimating, at the moment of writing this message, that about 50% of the registered participants will be able to attend the conference in-person, enjoying the atmosphere of the CCD congress center, the social events of the conference, and the many opportunities in Dublin. On the other side, virtual attendees will have the possibility to interact almost like they were in Dublin, thanks to a sophisticated virtual conference platform.

There are few important innovations this year. The most relevant is that ACL 2022 adopted a new reviewing process, based on "rolling review" (ARR), with the goal of coordinating and making more efficient the paper reviews of the ACL conferences. This initiative was shared with NAACL 2022, resulting in a coordinated effort. As a side effect of moving to ARR, we have been working on a new version of the software, called ACLPUB2, used to produce both the conference proceedings and the conference schedule. I would like to thank all the people who contributed to those achievements. Finally, this year we celebrate the 60th anniversary of the ACL conference. Thanks to the enthusiastic contributions of many organizations, coordinated by the Diversity and Inclusion co-chairs, we are preparing a very special initiative for our community, which, at the time of writing this message, is still secret and that will be disclosed during the opening of the conference.

I was very lucky to work together with three fantastic Program Chairs: Preslav Nakov, Smaranda Muresan and Aline Villaviciencio. I could not thank you more for the dedication and the capacity with which you have organized a very exciting scientific program and for the help in all the phases of the conference organization.

Thanks to the local organizers in Dublin, Andy Way and John Kelleher, and to the PCO, who managed the local organization in a period in which we have had very few certainties, and many more uncertainties.

We are extremely grateful to all sponsors for their continuing and generous support to help our conferences be very successful. Thank you to Chris Callison-Burch, the ACL Sponsorship Director, for managing the relations between the sponsors and ACL 2022.

I am also very grateful to the chairs of the previous years' conferences, who were always ready to help and to provide advice, contributing to the transmission, from year to year, of all the know-how and collective memory. Thanks to all the members of The ACL Executive Committee, they were always supportive, particularly when feedback on delicate issues was needed.

Many thanks to the senior area chairs, the area chairs, the reviewers, our workshop organizers, our tutorial instructors, the authors and presenters of papers, and the invited speakers.

ACL requires a long process, involving a large team of committed people. It is an honor for me to have coordinated such a team of talented people, who kindly volunteered their time to make this conference possible. I would like to thank the members of the organizing committee for their dedication and hard work, often under a tight schedule:

- Workshop Co-Chairs: Elena Cabrio, Sujian Li, Mausam;
- Tutorial Co-Chairs: Naoaki Okazaki, Yves Scherrer, Marcos Zampieri;
- Demo Co-Chairs: Valerio Basile, Zornitsa Kozareva, Sanja Štajner;
- Student Research Workshop Co-Chairs: Samuel Louvan, Brielen Madureira, Andrea Madotto;

- SRW Faculty Advisors: Cecile Paris, Siva Reddy, German Rigau;
- Publication Co-Chairs (also publication co-chairs for NAACL 2022): Danilo Croce, Ryan Cotterell, Jordan Zhang;
- Conference Handbook Chair: Marco Polignano;
- Diversity & Inclusion Co-chairs: Mona Diab, Martha Yifiru Tachbelie;
- Ethic advisor committee: Su Lin Blodgett, Christiane Fellbaum;
- Technical OpenReview Chair: Rodrigo Wilkens;
- Publicity and Social Media Co-chairs: Isabelle Augenstein, Emmanuele Chersoni, Diana Maynard, Soujanya Poria, Joel Tetreault;
- Local Arrangement Committee: Fiona McGillivray, Greg Carew, Laird Smith;
- Student Volunteer Coordinators: Filip Klubicka, Vasudevan Nedumpozhimana, Guodong Xie, Pintu Lohar;
- Internal Communications Chair: Marcely Boito Zanon.

Let me deserve a special thanks to Priscilla Rasmussen. She has been the pillar not only of this year's ACL, but of the ACL conferences for many years. She has offered her invaluable experience to the organizing committee, and her presence has always given us a pleasant sense of security.

Finally, I would like to thank all the participants, both in-person and virtual, who will be the main actors from May 22 to May 27, 2022. I am convinced that we will experience a fantastic conference, scientifically exciting and full of fond memories.

Welcome and hope you all enjoy the conference!

Bernardo Magnini (FBK, Italy)
ACL 2022 General Chair

Message from the Program Chairs

Welcome to the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022). ACL 2022 has a special historical significance, as this is the 60th Anniversary edition. It is also the first hybrid ACL conference after two years of a fully virtual format for ACL in 2020 and 2021 due to the COVID-19 pandemic. Finally, it is the first *ACL conference to fully embrace the ACL Rolling Review (ARR) as a reviewing process. Below, we discuss some of these changes and we highlight the exciting program that we have put together with the help from our community.

Using ARR for Reviewing

In coordination with the NAACL 2022 team and the ACL executive committee, we decided to fully adopt the ACL Rolling Review (ARR) as the only reviewing platform for ACL 2022. ARR is a new review system for *ACL conferences, where reviewing and acceptance of papers to publication venues is done in a two-step process: (*i*) centralized rolling review via ARR, and (*ii*) commitment to a publication venue, e.g., ACL 2022. The purpose of the ACL Rolling Review is to improve the efficiency and the turnaround of reviewing in *ACL conferences while keeping diversity (geographic and otherwise) and editorial freedom.

As ACL 2022 is the first conference to fully adopt the ARR review process, we worked very closely with ARR and we coordinated our efforts with the NAACL 2022 PC chairs. In particular, given the short distance between ACL 2022 and NAACL 2022, we allowed authors to commit their papers to ACL 2022 and simultaneously to submit a revision to ARR in January, which were eligible for NAACL 2022. We also joined ARR as Guest Editors-in-Chief (EiCs) to help with the September–November submissions to ARR, which primarily targeted ACL 2022. We worked together to integrate ARR and some of the conference workflows to ensure scaling up, and to maintain the quality and the timely processing of the submissions for November, and thus to guarantee that all papers submitted by the November 15, 2021 ARR deadline could be considered for ACL 2022 if the authors decided to commit them. This required making sure we had all reviews and meta-reviews ready in time, which we managed to achieve thanks to the combined efforts of the ARR and the ACL 2022 teams. We would also like to note that this is a community effort, and we are grateful for the support of the authors, the reviewers, the Action Editors (AEs), and the Senior Area Chairs (SACs), who have been constructively engaging and helping with ARR and ACL 2022.

Committing to ACL 2022

The commitment form for ACL 2022 asked the authors to provide a link to their paper in ARR: we asked for a link to the latest version of the paper that had reviews and a meta-review. The authors also needed to select an area (including the Special Theme area) they were submitting their paper to (this was needed as ACL 2022 had areas, while ARR did not). Finally, the authors were allowed to submit optional comments to the ACL 2022 Senior Area Chairs (SACs). Note that these comments were only visible to the SACs, and they were not sent to the reviewers or to the Action Editors: the rationale was that responding to reviewers and Action Editors should be handled in a response letter if the authors decided to do a resubmission in ARR, which is a completely different process than committing a paper to ACL 2022. These comments to the SACs were designed mainly to raise concerns about objective misunderstandings by the reviewers and/or by the Action Editor about the technical aspect of the paper that the authors believed might help the SACs in their decision-making process.

Areas While ARR did not have areas, ACL 2022 did: it had 23 areas, including the 22 areas from ACL 2021 plus our Special Theme. Our special theme was on "*Language Diversity: from Low-Resource to Endangered Languages*," to commemorate the 60th anniversary of ACL with the goal of reflecting and stimulating a discussion about how advances in computational linguistics and natural language processing can be used to promote language diversity from low-resource to endangered languages. We invited papers that discuss and reflect on the "role of the speech and language technologies in sustaining language use" (Bird, 2020) for the large variety of world languages with focus on under-resourced, indigenous, and/or endangered languages. We were interested in the challenges for developing and scaling up the current NLP technologies for the rich diversity of human languages and in the ethical, cultural, and policy implications of such technologies for local communities. We also have a best Theme paper award category.

Acceptance to ACL 2022

As ACL 2022 submissions in ARR, we count all papers from September, October, and November, which we advertised as ACL 2022 months, after removing all re-submissions and also nine papers that selected NAACL 2022 as a preferred venue (a total of 3,360 papers) + the papers from the May–August period that were actually committed to ACL 2022 and that were not resubmissions (a total of 18 papers), for a total of 3,378 papers.
This number is on par with the number of submissions to ACL 2021, which received 3,350 submissions. Subsequently, 1,918 papers were committed to ACL 2022 (i.e., 57%). After the review process, 701 papers (604 long and 97 short) were accepted into the main conference.

Acceptance Rates for the Main Conference

The quality of a conference is often perceived based on the acceptance rate of the papers submitted there, and thus it is important to have an acceptance rate that adequately represents the difficulty of publishing a paper in the conference. Given the adoption of ARR, it is also important to allow for consistency across various conferences. Thus, ACL 2022 (and NAACL 2022) adopted the following two ways of calculating the acceptance rates:

(a) *(Number of accepted papers at ACL 2022) / (Number of papers that selected ACL 2022 as the preferred venue in ARR or were committed to ACL 2022).* For ACL 2022, for the denominator we consider the 3,378 papers as explained above. Thus, the acceptance rate is 701 / 3,378 = 20.75% for the Main conference.

(b) *(Number of accepted papers at ACL 2022) / (Number of papers committed to ACL 2022).* For the denominator, we had 1,918 papers committed to ACL 2022, and thus, the acceptance rate is 701 / 1,918 = 36.54% for the Main conference.

Note that option (a) is closer to the way the acceptance rate was computed at previous *ACL conferences, where submitting and committing a paper was done in one step and papers were rarely withdrawn after the reviews, the meta-reviews, and the corresponding scores were released. However, one issue with this option for ACL 2022 was that indicating a preferred venue was only enabled starting with the October ARR submissions, and it was not available for earlier months. As mentioned above, we removed a small number of papers from our denominator that selected NAACL 2022 as a preferred venue in October and November (a total of 9 papers) and we considered the ARR submissions only for the months of September, October, and November, as these months were advertised in our CFP, plus any papers that were committed to ACL 2022 from earlier months (May–July) and which were also not resubmissions.
Option (b) yields a higher "acceptance rate", as many authors with low reviewing scores chose not to commit their paper to ACL 2022.

Best Paper Awards

From the committed ACL 2022 papers, we selected 32 papers as candidates for the following Best Paper awards, based on nominations by the Senior Area Chairs: Best Research Paper, Best Special Theme

Paper, Best Resource Paper, and Best Linguistic Insight Paper. These papers were assessed by the Best Paper Award Committee. The selected best papers will be presented in a dedicated plenary session for Best Paper Awards on May 24, 2022.

Findings of ACL 2022

Given the success of the Findings at EMNLP 2020 and 2021 and ACL-IJCNLP 2021, we also have Findings of ACL 2022 papers, which are papers that were not accepted for publication in the main conference, but nonetheless were assessed by the Program Committee as solid work with sufficient substance, quality, and novelty. A total of 361 papers were offered to be included in the Findings of ACL 2022. Given the two ways of computing acceptance rates described above, this results in a 10.68% acceptance rate in option (a), and 19.82% in option (b). Out of the 361 papers, 30 papers declined the offer, leading to 331 papers to be published in the Findings of ACL 2022. In order to increase the visibility of the Finding of ACL 2022 papers, we offered the authors of these 331 papers the possibility to present their work as a poster at ACL 2022, in addition to making a 6-minute or a 3-minute video to be included in the virtual conference site (for long and for short papers, respectively). The authors of 305 of the 331 papers accepted our invitation to present their work as a poster at ACL 2022.

TACL and Computational Linguistics

Continuing the tradition from previous years, ACL 2022 also features 43 articles that were published at the Transactions of the Association for Computational Linguistics (TACL) and 8 papers from the Computational Linguistics journal.

Keynote and Invited Speakers

Another highlight of our program are the keynotes, which we run in three different formats:

- **a keynote talk** by Angela Friederici (Max Planck Institute for Human Cognitive and Brain Sciences) on "*Language in the Human Brain*";

- **a keynote fire-side chat** on "*The Trajectory of ACL and the Next 60 years*" with Barbara Grosz (Harvard University) and Yejin Choi (University of Washington and Allen Institute for Artificial Intelligence), moderated by Rada Mihalcea (University of Michigan);

- **a keynote panel** on "How can we support linguistic diversity?" led by Steven Bird (Charles Darwin University), with panelists representing a variety of world languages, including (currently confirmed) Teresa Lynn (Irish), Robbie Jimerson (Seneca), Heather Long (Creole languages), and Manuel Mager (Wixaritari).

We further had two additional invited talk initiatives:

- **Spotlight Talks by Young Research Stars (STIRS)** by Eunsol Choi (University of Texas at Austin), Ryan Cotterell (ETH Zurich), Sebastian Ruder (Google, London), Swabha Swayamdipta (Allen Institute for AI), and Diyi Yang (Georgia Tech);

- **Next Big Ideas Talks** by Marco Baroni (Pompeu Fabra University), Eduard Hovy (The University of Melbourne and Carnegie Mellon University), Heng Ji (UIUC), Mirella Lapata (University of Edinburgh), Hang Li (Bytedance Technology), Dan Roth (University of Pennsylvania and Amazon), and Thamar Solorio (University of Houston).

Thank You

ACL 2022 is the result of a collaborative effort and a supportive community, and we want to acknowledge the efforts of so many people who have made significant efforts into the organization of ACL 2022! First of all, we would like to thank our Program Committee (the full list of names is quite long and it is included in the Program Committee pages of the Proceedings):

- Our awesome 82 Senior Area Chairs who were instrumental in every aspect of the review process, from liaising with ARR, to supporting the implementation of a two-stage reviewing system, recommending Action Editors and reviewers, working on paper acceptance, and nomination of best papers and outstanding reviewers. For all of them, this involved familiarizing themselves with a new protocol to accommodate the integration of ARR reviews and a new system, and for many of them, the scope of their responsibilities was equivalent to chairing a small conference.
- The 363 ARR Action Editors (from the June–November ARR cycles), who had the role of ACL 2022 Area Chairs interacting with reviewers, leading paper review discussions, and writing meta-reviews.
- The 2,323 ARR reviewers (from the June–November ARR cycles), who contributed for the ACL 2022 reviewing cycles, providing valuable feedback to the authors.
- The emergency ARR Action Editors and reviewers, who provided their support at the last minute to ensure a timely reviewing process.
- The amazing ARR team, who collaborated in the challenge of managing and implementing the ARR reviewing needed for the scale of ACL 2022. In particular, we acknowledge Amanda Stent and Goran Glavaš as Guest ARR Editors-in-Chief for ACL 2022, Graham Neubig as Guest ARR Chief Technical Officer for ACL 2022, and Sara Goggi as Guest ARR Editorial Manager for ACL 2022.

ACL 2022 counted on the contributions of many wonderful committees, including:

- Our Best Paper Selection Committee, who selected the best papers and the outstanding papers: Tim Baldwin, Kathleen McKeown, David Chiang, Min-Yen Kan, and Taro Watanabe.
- Our Ethics Advisory Committee, chaired by Christiane Fellbaum and Su Lin Blodgett, for their hard work to ensure that all the accepted papers addressed the ethical issues appropriately, under a very tight schedule and on a new platform.
- Our amazing Publication Chair Danilo Croce, our Handbook Chair Marco Polignano, the Technical OpenReview Chair Rodrigo Wilkens, and the Scheduler Chair Jordan Zhang, who jointly with the NAACL 2022 Publication Chair, Ryan Cotterell, made an enormous contribution to the community by implementing the integration scripts for generating the proceedings, the handbook and the schedule from the OpenReview platform.
- Our Publicity Chairs Isabelle Augenstein, Emmanuele Chersoni, Diana Maynard, Soujanya Poria, and Joel Tetreault, for their work on managing the communications on social media platforms.
- The Internal Communications Chair Marcely Boito Zanon for streamlining the processes.
- The wonderful Technical OpenReview Chair Rodrigo Wilkens, who went above and beyond to ensure that the typical ACL conference functionalities were translated to a new environment.

We would also like to thank many people who helped us with various software used for the conference:

- The ARR Tech team, in particular Sebastin Santy and Yoshitomo Matsubara, who served as Guest ARR Tech Team for ACL 2022.

- The OpenReview team, in particular Nadia L'Bahy, Celeste Martinez Gomez, and Melisa Bok, who helped to implement the integration of ARR as a reviewing platform for ACL 2022.

- The whole Underline team, in particular Sol Rosenberg, Jernej Masnec, Damira Mršić, and Mateo Antonic, who created a virtual site for the conference.

As Program chairs, we had to deal with many tasks, including handling new protocols and situations and a new conference management environment. We would not be able to complete these tasks without the advice from our colleagues, including

- Our fantastic General Chair Bernardo Magnini, who provided invaluable support and feedback throughout the whole process, including collaborating on the efforts to take on the challenge of reengineering the conference reviewing processes and pipeline.

- The Program Co-Chairs of NAACL 2022 Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, and the NAACL 2022 General Chair, Dan Roth, for collaborating in the challenge of coordinated adoption of ARR reviewing in a full scale for ACL 2022 and NAACL 2022.

- The Program Co-Chairs of previous editions of *ACL conferences, in particular the ACL-IJCNLP 2021 PC chairs Roberto Navigli, Fei Xia, and Wenjie Li, as well as the EMNLP 2021 PC chairs Lucia Specia, Scott Wen-tau Yih, and Xuanjing Huang for providing amazing guidance and support, and sharing their experience and answering our many questions, often on short notice.

- The ACL Executive Committee, especially Tim Baldwin (the ACL President), Rada Mihalcea (the ACL Past President), Shiqi Zhao (Secretary), Priscilla Rasmussen (Business Manager), and the members of the ACL executive committee for providing invaluable feedback and for helping us sort through various issues.

- The Computational Linguistics Editor-in-Chief Hwee Tou Ng, the TACL Editors-in-Chief Ani Nenkova and Brian Roark, and the TACL Editorial Assistant Cindy Robinson, for coordinating the Computational Linguistics and the TACL presentations at ACL 2022.

We would also like to thank all the authors who submitted/committed their work to ACL 2022. Although we were only able to accept a small percentage of the submissions, your hard work makes this conference exciting and our community strong. Our huge thanks goes to the *ACL communities for the kind and patient support during a year of major changes in our submission and reviewing processes.
Last, but not least, we thank our students, interns, postdocs, colleagues, and families for being so understanding and supportive during this intense year, and especially when we were swamped by countless conference deadlines and meetings. Our deepest gratitude is to all of you. We hope you will enjoy this 60th Anniversary edition of ACL.

Smaranda Muresan (Columbia University and Amazon AWS AI Labs, USA)
Preslav Nakov (Qatar Computing Research Institute, HBKU)
Aline Villavicencio (University of Sheffield, UK)

ACL 2022 Program Committee Co-Chairs

Message from the Local Chairs

Back in March 2020, just after the first COVID-19 lockdown, we submitted our bid for Dublin to host ACL 2022, conference that you are currently attending. In November 2020, we learned that our bid had been successful, which we were of course delighted to hear. Of course, at that stage – and at many points in between – we have wondered whether we would be able to meet face-to-face at all, and it is great that we are able to host you in the wonderful city of Dublin where we are privileged to live, as well as accommodating many of you online.

ACL is an opportunity to welcome not just our European friends and colleagues, but also those from farther afield. Ireland punches above its weight in the areas of NLP and Machine Learning, principally through the SFI-funded €100 million ADAPT Centre for Digital Content Technology, which comprises experts from 4 local Dublin universities as well as 4 further universities from across the country in a range of disciplines in AI. We have internationally renowned groups in machine translation, information retrieval, speech technology, parsing and grammar Induction, among others, so we believe it is appropriate that ACL is being held in our country for the first time. We are of course grateful to everyone who submitted a paper; whether your work was selected for presentation or not, if no-one had submitted, we wouldn't have had a conference. For those of you whose work was selected for presentation, many thanks for coming to Dublin, or for presenting online.

Along the way, we have been helped greatly by the General Chair Bernardo Magnini, and by Priscilla Rasmussen and others from the ACL executive team, to whom we are extremely thankful. However, by far the biggest thanks are due to Greg Carew and his team in Abbey Conference and Events for their professional support of the conference. You will have met them at registration, and they are available throughout the event to ensure your needs are met. We have been engaging with them for 2 years now on ACL, and for longer as they helped Andy host the MT Summit in 2019. We could not have made a better choice of PCO to assist us with all the requirements involved in hosting the best-regarded conference in our area. This has been a true partnership that has made this journey an enjoyable one.

We are also extremely grateful to Fáilte Ireland for their extremely generous support of this conference, and to our PostDocs Guodong Xie & Pintu Lohar (with Andy at DCU), and Vasudevan Nedumpozhimana & Filip Klubička (with John at TUD) for their huge efforts to recruit and manage the small army of student volunteers. Finally, we really hope that you all enjoy the conference, that you benefit from the excellent programme that has been assembled, and that you go away from here having made new friends. We are fortunate indeed that many of our very best friends are in the computational linguistics community, and we will try our very best to meet as many of you as possible during the event.

Andy Way (Dublin City University, Ireland)
John Kelleher (TU Dublin, Ireland)

Local Chairs, ACL 2022

Organizing Committee

General Chair

Bernardo Magnini, FBK, Italy

Program Chairs

Smaranda Muresan, Columbia University and Amazon AWS AI Labs, USA
Preslav Nakov, Qatar Computing Research Institute, HBKU, Qatar
Aline Villavicencio, University of Sheffield, UK

Local Organization Chairs

John Kelleher, TU Dublin, Ireland
Andy Way, Dublin City University, Ireland

Workshop Chairs

Elena Cabrio, Université Côte d'Azur, France
Sujian Li, Pekin University, China
Mausam, IIT Delhi, India

Tutorial Chairs

Luciana Benotti, National University of Córdoba, Argentina
Naoaki Okazaki, Tokyo Institute of Technology, Japan
Yves Scherrer, University of Helsinki, Finland
Marcos Zampieri, Rochester Institute of Technology, USA

Demo Chairs

Valerio Basile, University of Turin, Italy
Zornitsa Kozareva, Facebook AI Research, USA
Sanja Štajner, Symanto Research, Germany

Student Research Workshop Chairs

Samuel Louvan, FBK, Italy
Andrea Madotto, HKUST, Hong Kong
Brielen Madureira, University of Potsdam, Germany

Student Research Workshop: Faculty Advisors

Cecile Paris, CSIRO, Australia
Siva Reddy, McGill University, Canada
German Rigau, Basque Country University, Spain

Publicity and Social Media Chairs

Isabelle Augenstein, University of Copenhagen, Denmark
Emmanuele Chersoni, The Hong Kong Polytechnic University, Hong Kong
Diana Maynard, University of Sheffield, UK
Soujanya Poria, Singapore University of Technology, Singapore
Joel Tetreault, Dataminr, USA

Publication Chairs

Danilo Croce, University of Rome Tor Vergata, Italy
Ryan Cotterell, ETH Zürich, Switzerland
Jordan Zhang

Handbook Chair

Marco Polignano, University of Bari Aldo Moro, Italy

Technical OpenReview Chair

Rodrigo Wilkens, Université catholique de Louvain, Belgium

Conference App Chair

Pierluigi Cassotti, University of Bari Aldo Moro, Italy

Diversity and Inclusion Chairs

Mona Diab, Facebook AI Research & GWU, USA
Martha Yifiru Tachbelie, Addis Abada University, Ethiopia

Ethic Advisor Committee

Su Lin Blodgett, Microsoft Research Montréal, Canada
Christiane Fellbaum, Princeton University, USA

Student Volunteer Coordinators

Filip Klubička, ADAPT Centre, Ireland
Pintu Lohar, ADAPT Centre, Ireland
Vasudevan Nedumpozhimana, ADAPT Centre, Ireland
Guodong Xie, ADAPT Centre, Ireland

Internal Communications Chair

Marcely Boito Zanon, University of Avignon, France

Best Paper Selection Committee

Tim Baldwin, MBZUAI and The University of Melbourne, Australia
Kathleen McKeown, Columbia University, USA and Amazon AWS AI Labs
David Chiang, University of Notre Dame, USA

Min-Yen Kan, National University of Singapore, Singapore
Taro Watanabe, Nara Institute of Science and Technology, Japan

Guest ARR Editors-in-Chief for ACL 2022

Amanda Stent, Colby College, USA
Goran Glavaš, University of Mannheim, USA

Guest ARR Chief Technical Officer for ACL 2022

Graham Neubig, Carnegie Mellon University, USA

Guest ARR Editorial Manager for ACL 2022

Sara Goggi, CNR-ILC, Italy

Guest ARR Tech Team for ACL 2022

Yoshitomo Matsubara, UC Irvine, USA
Sebastin Santy, University of Washington, USA

Guest OpenReview Team for ACL 2022

Nadia L'Bahy, OpenReview
Celeste Martinez Gomez, OpenReview
Melisa Bok, OpenReview

Underline

Sol Rosenberg, Underline
Jernej Masnec, Underline
Damira Mršić, Underline
Mateo Antonic, Underline
Luka Šimić, Underline

Conference Advisor

Priscilla Rasmussen, ACL

Conference Registration

Nicole Ballard, Yes Events
Terah Shaffer, Yes Events

Local PCO

Greg Carew, Abbey
Fiona McGillivray, Abbey
Laird Smith, Abbey

Program Committee

Program Chairs

Smaranda Muresan, Columbia University and Amazon AWS AI Labs
Preslav Nakov, Qatar Computing Research Institute, HBKU
Aline Villavicencio, University of Sheffield

Computational Social Science and Cultural Analytics

Tanmoy Chakraborty, Indraprastha Institute of Information Technology Delhi
David Jurgens, University of Michigan
Diyi Yang, Georgia Institute of Technology

Dialogue and Interactive Systems

Srinivas Bangalore, Interactions LLC
Yun-Nung Chen, National Taiwan University
David Traum, University of Southern California
Dilek Hakkani-Tur, Amazon Alexa AI
Zhou Yu, Columbia University

Discourse and Pragmatics

Manfred Stede, Universität Potsdam
Junyi Jessy Li, University of Texas, Austin

Ethics and NLP

Saif M. Mohammad, National Research Council Canada
Malvina Nissim, University of Groningen

Generation

Claire Gardent, CNRS
Asli Celikyilmaz, Facebook AI Research
Chenghua Lin, University of Sheffield
Michael Elhadad, Ben Gurion University of the Negev

Information Extraction

Heng Ji, University of Illinois, Urbana-Champaign and Amazon Alexa AI
Marius Pasca, Google Research
Alan Ritter, Georgia Institute of Technology
Veselin Stoyanov, Facebook
Satoshi Sekine, RIKEN

Information Retrieval and Text Mining

Hang Li, Bytedance Technology
Marti Hearst, University of California Berkeley
Jing Jiang, Singapore Management University

Interpretability and Analysis of Models for NLP

Yonatan Belinkov, Technion, Technion
Anders Søgaard, Copenhagen University
Anna Rogers, University of Copenhagen
Hassan Sajjad, Qatar Computing Research Institute, HBKU

Language Grounding to Vision, Robotics and Beyond

William Yang Wang, UC Santa Barbara
Marie-Francine Moens, KU Leuven

Linguistic theories, Cognitive Modeling and Psycholinguistics

Frank Keller, The University of Edinburgh
Afra Alishahi, Tilburg University

Machine Learning for NLP

Mohit Bansal, University of North Carolina at Chapel Hill and Amazon Alexa AI
Nikolaos Aletras, University of Sheffield, University of Sheffield and Amazon
Andre Martins, Instituto Superior Técnico and Unbabel
Andreas Vlachos, Facebook and University of Cambridge
Kristina Toutanova, Google
Shafiq Joty, SalesForce and Nanyang Technological University

Machine Translation and Multilinguality

Taro Watanabe, Nara Institute of Science and Technology
Rico Sennrich, University of Zurich and University of Edinburgh
Francisco Guzmán, Facebook
Philipp Koehn, Facebook and Johns Hopkins University
Kenneth Heafield, The University of Edinburgh
Thamar Solorio, University of Houston

NLP Applications

Joel R. Tetreault, Dataminr
Karin Verspoor, Royal Melbourne Institute of Technology
Jimmy Lin, University of Waterloo
Horacio Saggion, Universitat Pompeu Fabra
Wei Gao, Singapore Management University
Beata Beigman Klebanov, Educational Testing Service

Phonology, Morphology and Word Segmentation

Ryan D Cotterell, ETH Zürich

Alexis Palmer, University of Colorado, Boulder

Question Answering

Mohit Iyyer, University of Massachusetts Amherst
Sanda Harabagiu, University of Texas at Dallas
Alessandro Moschitti, Amazon Alexa AI

Resources and Evaluation

Torsten Zesch, University of Duisburg-Essen
Agata Savary, Université Paris-Saclay
Katrin Erk, University of Texas, Austin
Pablo Gamallo, Universidad de Santiago de Compostela
Bonnie L. Webber, The University of Edinburgh

Semantics: Lexical

Carlos Ramisch, Aix Marseille University
Ekaterina Shutova, University of Amsterdam
Ivan Vulić, University of Cambridge and PolyAI Limited

Semantics: Sentence-level Semantics, Textual Inference and Other areas

Samuel R. Bowman, New York University
Goran Glavaš, University of Mannheim
Valeria de Paiva, Topos Institute
Renata Vieira, Universidade de Evora
Wei Lu, Singapore University of Technology and Design

Sentiment Analysis, Stylistic Analysis, and Argument Mining

Yulan He, The university of Warwick
Iryna Gurevych, TU Darmstadt
Roman Klinger, University of Stuttgart
Bing Liu, University of Illinois at Chicago

Special Theme

Emily M. Bender, University of Washington
Laurent Besacier, Naver Labs Europe
Steven Bird, Charles Darwin University and International Computer Science Institute

Speech and Multimodality

Grzegorz Chrupała, Tilburg University
Yang Liu, Amazon Alexa AI

Summarization

Kathleen McKeown, Columbia University and Amazon AWS AI Labs

Annie Louis, Google Research
Dragomir Radev, Yale University

Syntax: Tagging, Chunking and Parsing

Barbara Plank, IT University of Copenhagen
Joakim Nivre, Uppsala University

Action Editors

Zeljko Agic, Alan Akbik, Md Shad Akhtar, Firoj Alam, Nikolaos Aletras, Malihe Alikhani, Tanel Alumäe, Sophia Ananiadou, Antonios Anastasopoulos, Mark Anderson, Jacob Andreas, Xiang Ao, Marianna Apidianaki, Yuki Arase, Mikel Artetxe, Ehsaneddin Asgari, Giuseppe Attardi

Niranjan Balasubramanian, Timothy Baldwin, Miguel Ballesteros, David Bamman, Mohamad Hardyman Barawi, Jeremy Barnes, Loic Barrault, Roberto Basili, Ali Basirat, Jasmijn Bastings, Daniel Beck, Iz Beltagy, Luciana Benotti, Steven Bethard, Chandra Bhagavatula, Lidong Bing, Alexandra Birch, Steven Bird, Yonatan Bisk, Eduardo Blanco, Danushka Bollegala, Antoine Bosselut, Florian Boudin, Leonid Boytsov, Chloé Braud, Chris Brew, Wray Buntine

Elena Cabrio, Aoife Cahill, Andrew Caines, Ruken Cakici, Marie Candito, Yanan Cao, Ziqiang Cao, Cornelia Caragea, Xavier Carreras, Paula Carvalho, Andrew Cattle, Daniel Cer, Alessandra Cervone, Tanmoy Chakraborty, Muthu Kumar Chandrasekaran, Angel X Chang, Kai-Wei Chang, Snigdha Chaturvedi, Boxing Chen, Danqi Chen, Kehai Chen, Kuan-Yu Chen, Lei Chen, Yun-Nung Chen, Colin Cherry, Jackie CK Cheung, Hai Leong Chieu, Luis Chiruzzo, Jinho D. Choi, Monojit Choudhury, Khalid Choukri, Grzegorz Chrupała, Oana Cocarascu, Trevor Cohn, John M Conroy, Mathieu Constant, Caio Filippo Corro, Marta Ruiz Costa-jussà, Stefano Cresci, Aron Culotta

Giovanni Da San Martino, Raj Dabre, Walter Daelemans, Daniel Dakota, Dipanjan Das, Johannes Daxenberger, Gaël De Chalendar, Miryam De Lhoneux, Pascal Denis, Leon Derczynski, Barry Devereux, Mona T. Diab, Liang Ding, Georgiana Dinu, Jesse Dodge, Li Dong, Ruihai Dong, Yue Dong, Eduard Dragut, Kevin Duh, Nadir Durrani, Greg Durrett

Liat Ein-Dor, Michael Elhadad, Katrin Erk, Allyson Ettinger

Angela Fan, Anna Feldman, Naomi Feldman, Yang Feng, Yansong Feng, Raquel Fernández, Francis Ferraro, Elisabetta Fersini, Simone Filice, Mark Fishel, Annemarie Friedrich, Pascale Fung

Michel Galley, Matthias Gallé, Zhe Gan, Yang Gao, Marcos Garcia, Sebastian Gehrmann, Alborz Geramifard, Debanjan Ghosh, Goran Glavaš, Kyle Gorman, Jiatao Gu, Qing Gu, Honglei Guo, Qipeng Guo, Francisco Guzmán

Ivan Habernal, Christian Hardmeier, David Harwath, Luheng He, Yulan He, Zhongjun He, Daniel Hershcovich, Julia Hockenmaier, Enamul Hoque, Baotian Hu, Junjie Hu, Shujian Huang, Xuanjing Huang

Dmitry Ilvovsky, Kentaro Inui, Ozan Irsoy, Srini Iyer, Mohit Iyyer

Cassandra L Jacobs, Alon Jacovi, Kokil Jaidka, Hyeju Jang, Yangfeng Ji, Antonio Jimeno Yepes, Shafiq Joty, Preethi Jyothi

Sarvnaz Karimi, Shubhra Kanti Karmaker, Daisuke Kawahara, Daniel Khashabi, Jin-Dong Kim, Seokhwan Kim, Taeuk Kim, Judith Lynn Klavans, Roman Klinger, Hayato Kobayashi, Ekaterina Kochmar, Mamoru Komachi, Grzegorz Kondrak, Parisa Kordjamshidi, Amrith Krishna, Udo Kruschwitz, Marco Kuhlmann, Sumeet Kumar, Jonathan K Kummerfeld

Wai Lam, Zhenzhong Lan, Mark Last, Hady W. Lauw, Carolin Lawrence, John Lawrence, Alessandro Lenci, Lori Levin, Omer Levy, Mike Lewis, Jing Li, Junhui Li, Juntao Li, Junyi Jessy Li, Liangyou Li, Piji Li, Sujian Li, Tianrui Li, Wenjie Li, Zongxi Li, Constantine Lignos, Chenghua Lin, Dekang Lin, Marco Lippi, Pengfei Liu, Qun Liu, Wu Liu, Xuebo Liu, Yang Liu, Yang Liu, Zhiyuan Liu, Kyle Lo, Wei Lu, Thang Luong, Anh Tuan Luu

Wilson Ma, Craig MacDonald, Nitin Madnani, Andrea Madotto, Navonil Majumder, Prodromos Malakasiotis, Igor Malioutov, Thomas Mandl, Vukosi Marivate, Eugenio Martinez-Camara, Bruno Martins, Yuji Matsumoto, Mausam, David McClosky, Mahnoosh Mehrabani, Ivan Vladimir Meza Ruiz, Margot Mieskes, Makoto Miwa, Daichi Mochihashi, Saif M. Mohammad, Mohamed Morchid, David R Mortensen, Alessandro Moschitti, Lili Mou, Philippe Muller, Kenton Murray

Nona Naderi, Courtney Napoles, Shashi Narayan, Franco Maria Nardini, Tristan Naumann, Mark-Jan Nederhof, Vincent Ng, Dat Quoc Nguyen, Thien Huu Nguyen, Jan Niehues, Qiang Ning

Diarmuid O Seaghdha, Brendan O'Connor, Jose Ochoa-Luna, Kemal Oflazer, Maciej Ogrodniczuk, Alice Oh, Naoaki Okazaki, Manabu Okumura, Matan Orbach, Miles Osborne, Jessica Ouyang

Hamid Palangi, Ankur P Parikh, Joonsuk Park, Seong-Bae Park, Yannick Parmentier, Tommaso Pasini, Rebecca J. Passonneau, Viviana Patti, Haoruo Peng, Nanyun Peng, Gabriele Pergola, Fabio Petroni, Maxime Peyrard, Juan Pino, Emily Pitler, Edoardo Ponti, Simone Paolo Ponzetto, Kashyap Popat, Maja Popovic, Soujanya Poria, Vinodkumar Prabhakaran, Daniel Preotiuc-Pietro, Emily Prud'hommeaux

Tieyun Qian, Xipeng Qiu, Xiaojun Quan

Colin Raffel, Ganesh Ramakrishnan, Siva Reddy, Ines Rehbein, Roi Reichart, Xiang Ren, Yafeng Ren, Sebastian Riedel, Sara Rosenthal, Joseph Le Roux, Alla Rozovskaya, Attapol Rutherford

Mrinmaya Sachan, Benoît Sagot, Hassan Sajjad, Chinnadhurai Sankar, Maarten Sap, Nathan Schneider, Hinrich Schuetze, H. Schwartz, Lane Schwartz, Rico Sennrich, Minjoon Seo, Bei Shi, Tianze Shi, Lei Shu, Melanie Siegel, Kevin Small, Noah Smith, Luca Soldaini, Vivek Srikumar, Shashank Srivastava, Efstathios Stamatatos, Gabriel Stanovsky, Pontus Stenetorp, Amanda Stent, Veselin Stoyanov, Karl Stratos, Emma Strubell, Sara Stymne, Jinsong Su, Yu Su, Saku Sugawara, Jun Suzuki

Dima Taji, Zeerak Talat, Duyu Tang, Amalia Todirascu, Antonio Toral, Paolo Torroni, Kristina Toutanova, Amine Trabelsi, Trang Tran, Chen-Tse Tsai, Junichi Tsujii, Kewei Tu

Stefan Ultes

Olga Vechtomova, Giulia Venturi, Suzan Verberne, Yannick Versley, David Vilares, Serena Villata, Thuy Vu, Ivan Vulić, Yogarshi Vyas

Byron C Wallace, Xiaojun Wan, Jingjing Wang, Longyue Wang, Shuai Wang, Xin Eric Wang, Zhiguang Wang, Leo Wanner, Shinji Watanabe, Taro Watanabe, Bonnie L. Webber, Zhongyu Wei, Michael White, Alina Wróblewska, Lijun Wu

Tong Xiao, Deyi Xiong, Hainan Xu, Wei Xu

Rui Yan, Min Yang, Jin-Ge Yao, Wenpeng Yin, Koichiro Yoshino, Dian Yu, Jianfei Yu, Kai Yu, Mo Yu, Tao Yu, François Yvon

Marcos Zampieri, Fabio Massimo Zanzotto, Luke Zettlemoyer, Justine Zhang, Weinan Zhang, Xiangliang Zhang, Xingxing Zhang, Yi Zhang, Yue Zhang, Zhe Zhang, Xiaoqing Zheng, Michael Zock

ARR reviewers

Micheal Abaho, Ahmed Abdelali, Mostafa Abdou, Muhammad Abdul-Mageed, Omri Abend, Abdalghani Abujabal, Lasha Abzianidze, Manoj Acharya, Heike Adel, David Ifeoluwa Adelani, Somak Aditya, Vaibhav Adlakha, Stergos D. Afantenos, Sachin Agarwal, Vibhav Agarwal, Rodrigo Agerri, Manex Agirrezabal, Ameeta Agrawal, Priyanka Agrawal, Sweta Agrawal, Gustavo Aguilar, Roee Aharoni, Wasi Uddin Ahmad, Benyamin Ahmadnia, Aman Ahuja, Chaitanya Ahuja, Kabir Ahuja, Xi Ai, Laura Aina, Akiko Aizawa, Alan Akbik, Md Shad Akhtar, Nader Akoury, Ekin Akyürek, Ozge Alacam, Firoj Alam, Mehwish Alam, Chris Alberti, Georgios Alexandridis, David Alfter, Bashar Alhafni, Raquel G. Alhama, Tariq Alhindi, Hamed Alhoori, Hassan Alhuzali, Mohammad Aliannejadi, Afra Alishahi, Tamer Alkhouli, Emily Allaway, Miguel A. Alonso, Sawsan Alqahtani, Emily Alsentzer, Milad Alshomary, Christoph Alt, Tanel Alumäe, Fernando Alva-Manchego, Rami Aly, Maxime Amblard, Prithviraj Ammanabrolu, Reinald Kim Amplayo, Chantal Amrhein, Aixiu An, Guozhen An, Ashish Anand, Sophia Ananiadou, Raviteja Anantha, Antonios Anastasopoulos, Carolyn Jane Anderson, Nicholas Andrews, Ion Androutsopoulos, Gabor Angeli, Diego Antognini, Kaveri Anuranjana, Emilia Apostolova, Jun Araki, Rahul Aralikatte, Eiji Aramaki, Yuki Arase, Arturo Argueta, Mozhdeh Ariannezhad, Ignacio Arroyo-Fernández, Katya Artemova, Yoav Artzi, Masayuki Asahara, Akari Asai, Meysam Asgari, Elliott Ash, Zhenisbek Assylbekov, Duygu Ataman, Dennis Aumiller, Eleftherios Avramidis, Parul Awasthy, Hosein Azarbonyad, Wilker Aziz

Rohit Babbar, Sanghwan Bae, Ebrahim Bagheri, Dzmitry Bahdanau, Ashutosh Baheti, Fan Bai, He Bai, Yu Bai, JinYeong Bak, Vidhisha Balachandran, Mithun Balakrishna, Anusha Balakrishnan, Niranjan Balasubramanian, Ioana Baldini, Livio Baldini Soares, Kalika Bali, Nicolae Banari, Juan M Banda, Pratyay Banerjee, Sameer Bansal, Trapit Bansal, Forrest Sheng Bao, Hangbo Bao, Jianzhu Bao, Junwei Bao, Siqi Bao, Yu Bao, Zuyi Bao, Ankur Bapna, Roy Bar-Haim, Edoardo Barba, Francesco Barbieri, Denilson Barbosa, M Saiful Bari, Ken Barker, Gianni Barlacchi, Jeremy Barnes, Maria Barrett, Valentin Barriere, James Barry, Max Bartolo, Pierpaolo Basile, Valerio Basile, Somnath Basu Roy Chowdhury, John A. Bateman, Riza Batista-Navarro, Anil Batra, Khuyagbaatar Batsuren, Daniel Bauer, Timo Baumann, Rachel Bawden, Kathy Baxter, Tilman Beck, Lee Becker, Lisa Beinborn, Ahmad Beirami, Giannis Bekoulis, Núria Bel, Eric Bell, Gábor Bella, Meriem Beloucif, Iz Beltagy, Eyal Ben-David, Emily M. Bender, Michael Bendersky, Luisa Bentivogli, Adrian Benton, Jonathan Berant, Alexandre Berard, Gábor Berend, Taylor Berg-Kirkpatrick, Toms Bergmanis, Rafael Berlanga, Delphine Bernhard, Dario Bertero, Laurent Besacier, Chandra Bhagavatula, Rishabh Bhardwaj, Aditya Bhargava, Suma Bhat, Parminder Bhatia, Sumit Bhatia, Kasturi Bhattacharjee, Pushpak Bhattacharyya, Satwik Bhattamishra, Shruti Bhosale, Rajarshi Bhowmik, Bin Bi, Wei Bi, Federico Bianchi, Laura Biester, Yi Bin, Lidong Bing, Philippe Blache, Fred Blain, Eduardo Blanco, Terra Blevins, Rexhina Blloshmi, Jelke Bloem, Michael Bloodgood, Valts Blukis, Ben Bogin, Nikolay Bogoychev, Ondrej Bojar, Gemma Boleda, Danushka Bollegala, Marcel Bollmann, Valeriia Bolotova, Daniele Bonadiman, Francis Bond, Claudia Borg, Mihaela Bornea, Aurélien Bossard, Antoine Bosselut, Robert Bossy, Nadjet Bouayad-Agha, Florian Boudin, Zied Bouraoui, Samuel R. Bowman, Jordan Lee Boyd-Graber, Johan Boye, Kristy Elizabeth Boyer, Faeze Brahman, Arthur Brazinskas, Thomas Brochhagen, Samuel Broscheit, Thomas Brovelli, Christopher Bryant, Paweł Budzianowski, Emanuele Bugliarello, Wray Buntine, Joan Byamugisha, Bill Byrne

Sky CH-Wang, Subalalitha CN, Elena Cabrio, Avi Caciularu, Samuel Cahyawijaya, Deng Cai, Han Cai, Hengyi Cai, Jon Cai, Pengshan Cai, Yi Cai, Andrew Caines, Agostina Calabrese, Iacer Calixto, Jose Camacho-Collados, Erik Cambria, Oana-Maria Camburu, Giovanni Campagna, Leonardo Campillos-Llanos, Daniel F Campos, Jon Ander Campos, Marie Candito, Jie Cao, Juan Cao, Kris Cao, Meng Cao, Qingqing Cao, Qingxing Cao, Ruisheng Cao, Steven Cao, Yixin Cao, Yu Cao, Yuan Cao, Yue Cao, Yunbo Cao, Annalina Caputo, Doina Caragea, Dallas Card, Ronald Cardenas, Rémi Cardon, Danilo Carvalho, Tommaso Caselli, Justine Cassell, Vittorio Castelli, Giuseppe Castellucci, Thiago Castro Ferreira, Paulo Cavalin, Christophe Cerisara, Alessandra Cervone, Arun Tejasvi Chaganty, Soumen Chakrabarti, Abhisek Chakrabarty, Tuhin Chakrabarty, Tanmoy Chakraborty, Bharathi Raja Chakravarthi, Ilias Chalkidis, Jon Chamberlain, Nathanael Chambers, Angel X Chang, Baobao Chang, Haw-Shiuan Chang, Jonathan P. Chang, Serina Chang, Xuankai Chang, Lidia S. Chao, WenHan Chao, Akshay Chaturvedi, Aditi Chaudhary, Vishrav Chaudhary, Wanxiang Che, Bei Chen, Bo Chen, Chenhua Chen, Chung-Chi Chen, Danqi Chen, Daoyuan Chen, Guanhua Chen, Guanyi Chen, Hanjie Chen, Hongshen Chen, Howard Chen, Huimin Chen, Jiaze Chen, Jifan Chen, John Chen, Kehai Chen, Lei Chen, Lin Chen, Long Chen, Lu Chen, Luoxin Chen, Maximillian Chen, Mei-Hua Chen, Meng Chen, Mingda Chen, Minhua Chen, Muhao Chen, Pei Chen, Pinzhen Chen, Qian Chen, Qianglong Chen, Qingcai Chen, Sanxing Chen, Shizhan Chen, Tao Chen, Wang Chen, Wei-Fan Chen, Wenhu Chen, Wenliang Chen, Wenqing Chen, Xilun Chen, Xinchi Chen, Xiusi Chen, Xiuying Chen, Yen-Chun Chen, Yubo Chen, Yue Chen, Yufeng Chen, Yulong Chen, Yun Chen, Yun-Nung Chen, Yunmo Chen, Zhi Chen, Zhihong Chen, Zhuang Chen, Zhumin Chen, Zhuohao Chen, Fei Cheng, Hao Cheng, Jianpeng Cheng, Liying Cheng, Lu Cheng, Minhao Cheng, Pengxiang Cheng, Pengyu Cheng, Weiwei Cheng, Yong Cheng, Yu Cheng, Emmanuele Chersoni, Ethan A Chi, Ta-Chung Chi, Zewen Chi, Yew Ken Chia, David Chiang, Ting-Rui Chiang, Patricia Chiril, Francisco Javier Chiyah-Garcia, Jaemin Cho, Sangwoo Cho, Won Ik Cho, Eleanor Chodroff, Eunsol Choi, Jaesik Choi, Jinho D. Choi, Seungtaek Choi, Shamil Chollampatt, Jaegul Choo, Leshem Choshen, Prafulla Kumar Choubey, Monojit Choudhury, Jishnu Ray Chowdhury, Md Faisal Mahbub Chowdhury, Shammur Absar Chowdhury, Christos Christodoulopoulos, Fenia Christopoulou, Alexandra Chronopoulou, Chenhui Chu, Christopher Chu, Zewei Chu, Tat-Seng Chua, Jin-Woo Chung, Yi-Ling Chung, Kenneth Church, Abu Nowshed Chy, Mark Cieliebak, Manuel Rafael Ciosici, Volkan Cirik, Christopher Clark, Elizabeth Clark, Kevin Clark, Miruna Clinciu, Louis Clouatre, Trevor Cohen, Jeremy R. Cole, Marcus D. Collins, Simone Conia, Mathieu Constant, Danish Contractor, Robin Cooper, Anna Corazza, Luciano Del Corro, Ryan D Cotterell, Josep Crego, Danilo Croce, Paul A. Crook, James Cross, Fermín L. Cruz, Heriberto Cuayahuitl, Lei Cui, Leyang Cui, Shaobo Cui, Yiming Cui, Washington Cunha, Anna Currey, Tonya Custis, Erion Çano

Luis Fernando D'Haro, Jennifer D'Souza, Giovanni Da San Martino, Raj Dabre, Deborah A. Dahl, Damai Dai, Falcon Z Dai, Hongliang Dai, Wenliang Dai, Xiang Dai, Xinyu Dai, Yinpei Dai, Siddharth Dalmia, Sandipan Dandapat, Ankit Dangi, Marina Danilevsky, Verna Dankers, Anubrata Das, Rajarshi Das, Sarthak Dash, Pradeep Dasigi, Debajyoti Datta, Hal Daumé Iii, Sam Davidson, Brian Davis, Ernest Davis, Gaël De Chalendar, Christine De Kock, Kordula De Kuthy, Miryam De Lhoneux, Marie-Catherine De Marneffe, Gerard De Melo, José G. C. De Souza, Iria De-Dios-Flores, Steve DeNeefe, Alok Debnath, Mathieu Dehouck, Flor Miriam Plaza Del Arco, Marco Del Tredici, Agustín D. Delgado, Louise Deléger, David Demeter, Çağatay Demiralp, Yang Deng, Yuntian Deng, Zhongfen Deng, Tejaswini Deoskar, Jan Milan Deriu, Franck Dernoncourt, Tim Dettmers, Daniel Deutsch, Sunipa Dev, Joseph Dexter, Kuntal Dey, Bhuwan Dhingra, Luigi Di Caro, Barbara Di Eugenio, Shizhe Diao, Gaël Dias, Chenchen Ding, Haibo Ding, Kaize Ding, Liang Ding, Ning Ding, Shuoyang Ding, Xiao Ding, Stefanie Dipper, Nemanja Djuric, Ngoc Bich Do, Simon Dobnik, Jesse Dodge, Charles Dognin, Miguel Domingo, Lucia Donatelli, Domenic Donato, Li Dong, MeiXing Dong, Qian Qian Dong, Yue Dong, Bonaventure F. P. Dossou, Longxu Dou, Zi-Yi Dou, Doug Downey, A. Seza Doğruöz, Mark Dras, Markus Dreyer, Rotem Dror, Andrew Drozdov, Jingfei Du, Jinhua Du, Lan Du, Li Du, Mengnan Du, Pan Du, Wanyu Du, Xinya Du, Yupei Du, Junwen Duan, Xiangyu Duan, Kumar Avinava Dubey, Pablo Duboue, Philipp Dufter, Jonathan Dunn, Gérard M Dupont, Ondrej Dusek, Ritam Dutt, Subhabrata Dutta, Chris Dyer, Nouha Dziri, Hervé Déjean

Abteen Ebrahimi, Aleksandra Edwards, Steffen Eger, Markus Egg, Koji Eguchi, Yo Ehara, Vladimir Eidelman, Bryan Eikema, Jacob Eisenstein, Asif Ekbal, Wassim El-Hajj, Aparna Elangovan, Yanai Elazar, Heba Elfardy, Michael Elhadad, AbdelRahim A. Elmadany, Micha Elsner, Denis Emelin, Guy Emerson, Akiko Eriguchi, Liana Ermakova, Patrick Ernst, Carlos Escolano, Arash Eshghi, Ramy Eskander, Cristina España-Bonet, Luis Espinosa-Anke, Kawin Ethayarajh, Allyson Ettinger, Kilian Evang, Ben Eyal

Alexander Fabbri, Marzieh Fadaee, Tiziano Fagni, Farzane Fakhrian, Neele Falk, Tobias Falke, Chuang Fan, Feifan Fan, Kai Fan, Lu Fan, Wei Fang, Yimai Fang, Yuwei Fang, Adam Faulkner, Maryam Fazel-Zarandi, Amir Feder, Hao Fei, Nils Feldhus, Naomi Feldman, Mariano Felice, Jiazhan Feng, Shaoxiong Feng, Shi Feng, Shi Feng, Xiachong Feng, Zhangyin Feng, Manos Fergadiotis, James Ferguson, Patrick Fernandes, Raquel Fernández, Daniel Fernández-González, Elisa Ferracane, Francis Ferraro, Besnik Fetahu, Oluwaseyi Feyisetan, Alejandro Figueroa, Simone Filice, Catherine Finegan-Dollak, Orhan Firat, Nicholas FitzGerald, Margaret M. Fleck, Lucie Flek, Antske Fokkens, Marina Fomicheva, José A.r. Fonollosa, Marco Fonseca, Tommaso Fornaciari, Paula Fortuna, Eric Fosler-Lussier, George Foster, Jennifer Foster, Mary Ellen Foster, Anette Frank, Stella Frank, Thomas François, Alexander Fraser, Kathleen C. Fraser, Marjorie Freedman, Dayne Freitag, Markus Freitag, Lea Frermann, Daniel Fried, Guohong Fu, Jie Fu, Peng Fu, Qiankun Fu, Tsu-Jui Fu, Zuohui Fu, Yoshinari Fujinuma, Atsushi Fujita, Kotaro Funakoshi, Adam Funk, Richard Futrell, Michael Färber

Devi G, Matteo Gabburo, Saadia Gabriel, David Gaddy, Marco Gaido, Andrea Galassi, Mark Gales, Boris Alexandrovich Galitsky, Ygor Gallina, Diana Galvan, Björn Gambäck, Leilei Gan, Yujian Gan, Zhe Gan, Kuzman Ganchev, Sudeep Gandhe, Balaji Ganesan, Rashmi Gangadharaiah, Varun Gangal, Revanth Gangi Reddy, Debasis Ganguly, Ge Gao, Jun Gao, Shen Gao, Tianyu Gao, Wei Gao, Yang Gao, Yanjun Gao, Yifan Gao, Yingbo Gao, Utpal Garain, Cristina Garbacea, Diego Garcia-Olano, Matt Gardner, Sarthak Garg, Siddhant Garg, Dan Garrette, Aina Garí Soler, Kiril Gashteovski, Albert Gatt, Manas Gaur, Eric Gaussier, Dipesh Gautam, Yubin Ge, Sebastian Gehrmann, Michaela Geierhos, Ruiying Geng, Shijie Geng, Xinwei Geng, Xiubo Geng, Ariel Gera, Mor Geva, Hamidreza Ghader, Demian Gholipour Ghalandari, Sarik Ghazarian, Mozhdeh Gheini, Deepanway Ghosal, Deepanway Ghosal, Debanjan Ghosh, Sayan Ghosh, Soumitra Ghosh, Sourav Ghosh, Daniel Gildea, Salvatore Giorgi, Voula Giouli, Adrià de Gispert, Mario Giulianelli, Michael Glass, Goran Glavaš, Alfio Gliozzo, Pranav Goel, Vaibhava Goel, Nazli Goharian, Tejas Gokhale, Elizaveta Goncharova, Heng Gong, Hongyu Gong, Karthik Gopalakrishnan, Philip John Gorinski, Matthew R. Gormley, Koustava Goswami, Akhilesh Deepak Gotmare, Isao Goto, Cyril Goutte, Edward Gow-Smith, Kartik Goyal, Naman Goyal, Pawan Goyal, Tanya Goyal, Mario Graff, Christophe Gravier, Yulia Grishina, Milan Gritta, Loïc Grobol, Dagmar Gromann, Roman Grundkiewicz, Jia-Chen Gu, Jing Gu, Yue Gu, Jian Guan, Saiping Guan, Yi Guan, Marco Guerini, Lin Gui, Tao Gui, Vincent Guigue, Liane Guillou, Camille Guinaudeau, Kalpa Gunaratna, Chulaka Gunasekara, Tunga Gungor, Jiang Guo, Jiaqi Guo, Junliang Guo, Ruocheng Guo, Yinpeng Guo, Yuhang Guo, Zhijiang Guo, Arpit Gupta, Arshit Gupta, Nitish Gupta, Prakhar Gupta, Shashank Gupta, Sonal Gupta, Vivek Gupta, Izzeddin Gur, Suchin Gururangan, Joakim Gustafson, Ximena Gutierrez-Vasques, Carlos Gómez-Rodríguez

Jung-Woo Ha, Nizar Habash, Ivan Habernal, Kais Haddar, Christian Hadiwinoto, Reza Haf, Michael Hahn, Zhen Hai, Huda Hakami, Dilek Hakkani-Tur, Kishaloy Halder, Jiale Han, Jiawei Han, Namgi Han, Rujun Han, Ting Han, Wenjuan Han, Xianpei Han, Xiaochuang Han, Abram Handler, Viktor Hangya, Greg Hanneman, Jie Hao, Yaru Hao, Momchil Hardalov, Mareike Hartmann, Thomas Hartvigsen, Sadid A. Hasan, Peter Hase, Chikara Hashimoto, Nabil Hathout, Robert D. Hawkins, Hiroaki Hayashi, Katsuhiko Hayashi, Yoshihiko Hayashi, Shirley Anugrah Hayati, Devamanyu Hazarika, Hangfeng He, Jiangen He, Junxian He, Keqing He, Liang He, Luheng He, Shizhu He, Tianxing He, Wanwei He, Xuanli He, Zhongjun He, Marti Hearst, Michael Heck, Behnam Hedayatnia, Benjamin Heinzerling, Matthew Henderson, Iris Hendrickx, Leonhard Hennig, Sophie Henning, Daniel Hershcovich, Jonathan Herzig, Jack Hessel, John Hewitt, Ryuichiro Higashinaka, Swapnil Hingmire, Tsutomu Hirao, Tatsuya Hiraoka, Cuong Hoang, Hieu Hoang, Johannes Hoffart, Valentin Hofmann, Chris Hokamp, Eben Holderness, Nora Hollenstein, Ari Holtzman, Takeshi Homma, Ukyo Honda, Pengfei Hong, Mark Hopkins, Helmut Horacek, Md Mosharaf Hossain, Nabil Hossain, Mohammad Javad Hosseini, Feng Hou, Lei Hou, Yufang Hou, Yutai Hou, Dirk Hovy, David M Howcroft, Estevam Hruschka, Shu-Kai Hsieh, Chao-Chun Hsu, Chun-Nan Hsu, I-Hung Hsu, Wei-Ning Hsu, Phu Mon Htut, Baotian Hu, Chi Hu, Guangneng Hu, Huang Hu, Jennifer Hu, Jinyi Hu, Junjie Hu, Linmei Hu, Minghao Hu, Pengwei Hu, Po Hu, Renfen Hu, Wei Hu, Yue Hu, Zhe Hu, Ziniu Hu, Xinyu Hua, Yiqing Hua, Chao-Wei Huang, Chenyang Huang, Chieh-Yang Huang, Danqing Huang, Fei Huang, Haoran Huang, He Huang, Hen-Hsen Huang, Heyan Huang, Jiaji Huang, Jie Huang, Jimmy Huang, Jing Huang, Kuan-Hao Huang, Kung-Hsiang Huang, Lifu Huang, Minlie Huang, Quzhe Huang, Ruihong Huang, Shujian Huang, Siyu Huang, Xiaolei Huang, Xinting Huang, Xuancheng Huang, Zhen Huang, Zhongqiang Huang, Ziming Huang, Patrick Huber, Binyuan Hui, Kai Hui, Dieuwke Hupkes, Ben Hutchinson, Jena D. Hwang, Sung Ju Hwang, Ali Hürriyetoğlu

Ignacio Iacobacci, Georgiana Ifrim, Oana Ignat, Ryu Iida, Gabriel Ilharco, Filip Ilievski, Nikolai Ilinykh, Irina Illina, Dmitry Ilvovsky, Kenji Imamura, Oana Inel, Naoya Inoue, Radu Tudor Ionescu, Daphne Ippolito, Hitoshi Isahara, Tatsuya Ishigaki, Etsuko Ishii, Tunazzina Islam, Hayate Iso, Dan Iter, Itay Itzhak, Julia Ive, Tomoya Iwakura, Kenichi Iwatsuki, Rishabh K Iyer, Srini Iyer, Gautier Izacard

Aaron Jaech, Sarthak Jain, Masoud Jalili Sabet, Abhik Jana, Hyeju Jang, Tommi Jauhiainen, Sébastien Jean, Sungho Jeon, Minwoo Jeong, Yacine Jernite, Kevin Jesse, Rahul Jha, Harsh Jhamtani, Feng Ji, Zongcheng Ji, Chen Jia, Renee Jia, Robin Jia, Ruipeng Jia, Yuxiang Jia, Ping Jian, Daxin Jiang, Haoming Jiang, Hongfei Jiang, Meng Jiang, Ming Jiang, Nan Jiang, Nanjiang Jiang, Wenbin Jiang, Xin Jiang, Yong Jiang, Zhengbao Jiang, Zhuolin Jiang, Zhuoren Jiang, Zhuoxuan Jiang, Wenxiang Jiao, Zhanming Jie, Antonio Jimeno Yepes, Di Jin, Hailong Jin, Hanqi Jin, Lisa Jin, Peng Jin, Xiaolong Jin, Zhijing Jin, Baoyu Jing, Yohan Jo, Richard Johansson, Melvin Johnson, Erik Jones, Gareth J. F. Jones, Siddhartha Jonnalagadda, Aditya Joshi, Dhanya Jothimani, Shafiq Joty, Xincheng Ju, Jaap Jumelet, Heewoo Jun, David Jurgens, Prathyusha Jwalapuram

Jad Kabbara, Indika Kahanda, Sylvain Kahane, Ivana Kajic, Mihir Kale, Oren Kalinsky, Aikaterini-Lida Kalouli, Ehsan Kamalloo, Hidetaka Kamigaito, Jaap Kamps, Min-Yen Kan, Hiroshi Kanayama, Nikhil Kandpal, Masahiro Kaneko, Dongyeop Kang, Minki Kang, Diptesh Kanojia, Evangelos Kanoulas, Jiun-Yu Kao, Pavan Kapanipathi, Georgi Karadzhov, Alina Karakanta, Giannis Karamanolakis, Siddharth Karamcheti, Mladen Karan, Börje F. Karlsson, Sanjeev Kumar Karn, Jungo Kasai, Omid Kashefi, Yosuke Kashiwagi, Zdeněk Kasner, Nora Kassner, Denys Katerenchuk, Divyansh Kaushik, Pride Kavumba, Anna Kazantseva, Hideto Kazawa, Ashkan Kazemi, Abe Kazemzadeh, Pei Ke, Zixuan Ke, Chris Kedzie, Katherine A. Keith, Yova Kementchedjhieva, Brendan Kennedy, Casey Kennington, Tom Kenter, Daniel J Kershaw, Santosh Kesiraju, Salam Khalifa, Dinesh Khandelwal, Urvashi Khandelwal, Simran Khanuja, Mitesh M Khapra, Eugene Kharitonov, Daniel Khashabi, Mikhail Khodak, Tushar Khot, Johannes Kiesel, Halil Kilicoglu, Byeongchang Kim, Dong-Jin Kim, Dongkwan Kim, Doo Soon Kim, Gene Louis Kim, Geonmin Kim, Gunhee Kim, Gyuwan Kim, Hyounghun Kim, Hyunwoo Kim, Jihyuk Kim, Jin-Dong Kim, Joo-Kyung Kim, Jooyeon Kim, Jung-jae Kim, Juyong Kim, Kang-Min Kim, Seokhwan Kim, Yeachan Kim, Yoon Kim, Yunsu Kim, Milton King, Tracy Holloway King, Christo Kirov, Nikita Kitaev, Hirokazu Kiyomaru, Shun Kiyono, Judith Lynn Klavans, Ayal Klein, Bennett Kleinberg, Jan-Christoph Klie, Mateusz Klimaszewski, Miyoung Ko, Hideo Kobayashi, Sosuke Kobayashi, Thomas H Kober, Jordan Kodner, Svetla Peneva Koeva, Mare Koit, Noriyuki Kojima, Alexander Koller, Keshav Kolluru, Mamoru Komachi, Rik Koncel-Kedziorski, Lingkai Kong, Luyang Kong, Valia Kordoni, Yuta Koreeda, Mandy Barrett Korpusik, Katsunori Kotani, Lili Kotlerman, Fajri Koto, Venelin Kovatchev, Josip Krapac, Sebastian Krause, Elisa Kreiss, Ralf Krestel, Julia Kreutzer, Florian L. Kreyssig, Kalpesh Krishna, Nikhil Krishnaswamy, Reno Kriz, Canasai Kruengkrai, Udo Kruschwitz, Germàn Kruszewski, Alexander Ku, Lun-Wei Ku, Marco Kuhlmann, Mayank Kulkarni, Sayali Kulkarni, Vivek Kulkarni, Artur Kulmizev, Devang Kulshreshtha, Ashutosh Kumar, Dhruv Kumar, Sachin Kumar, Sawan Kumar, Shankar Kumar, Varun Kumar, Vishwajeet Kumar, Anoop Kunchukuttan, Souvik Kundu, Shuhei Kurita, Kemal Kurniawan, Sadao Kurohashi, Robin Kurtz, Andrey Kutuzov

Peifeng LI, Matthieu Labeau, Faisal Ladhak, Nikolaos Lagos, Cheng-I Lai, Viet Dac Lai, Yuxuan Lai, Yash Kumar Lal, Divesh Lala, John P. Lalor, Tsz Kin Lam, Wai Lam, Matthew Lamm, Vasileios Lampos, Gerasimos Lampouras, Man Lan, Yanyan Lan, Yunshi Lan, Lukas Lange, Ni Lao, Guy Lapalme, Egoitz Laparra, Mirella Lapata, Gabriella Lapesa, Ekaterina Lapshinova-Koltunski, Stefan Larson, Jey Han Lau, Anne Lauscher, Alberto Lavelli, John Lawrence, Dawn Lawrie, Hung Le, Phong Le, Andrew Lee, Dongkyu Lee, Fei-Tzin Lee, Hung-yi Lee, Hwanhee Lee, Hwaran Lee, Hyunju Lee, I-Ta Lee, Ji-Ung Lee, Jinhyuk Lee, Katherine Lee, Kenton Lee, Kyungjae Lee, Mina Lee, Moontae Lee, Nayeon Lee, Roy Ka-Wei Lee, Sang-Woo Lee, Seolhwa Lee, Taesung Lee, Young-Suk Lee, Artuur Leeuwenberg, Els Lefever, Jie Lei, Tao Lei, Wenqiang Lei, Zeyang Lei, Jochen L. Leidner, Alessandro Lenci, Yichong Leng, Haley Lepp, Piyawat Lertvittayakumjorn, Guy Lev, Lori Levin, Gina-Anne Levow, Bai Li, Baoli Li, Bei Li, Binyang Li, Bo Li, Bowen Li, Chen Li, Chen Li, Chenliang Li, Chenliang Li, Chunyuan Li, Dianqi Li, Dingcheng Li, Dongfang Li, Fei Li, Haizhou Li, Haoran Li, Hongyu Li, Huayang Li, Irene Li, Jialu Li, Jicheng Li, Jinchao Li, Jing Li, Jing Li, Jingye Li, Juanzi Li, Juncheng B Li, Junyi Jessy Li, Lei Li, Lin Li, Lucy Li, Manling Li, Miao Li, Minglei Li, Peng Li, Piji Li, Qi Li, Quanzhi Li, Ruizhe Li, Shang-Wen Li, Shaohua Li, Sheng Li, Shuangyin Li, Si Li, Tao Li, Wei Li, Xiang Lisa Li, Xiang Li, Xiang Lorraine Li, Xiangci Li, Xiaonan Li, Xin Li, Xinjian Li, Xintong Li, Xiujun Li, Yaliang Li, Yanran Li, Yanzeng Li, Yaoyiran Li, Yingjie Li, Yingya Li, Yinqiao Li, Yitong Li, Yitong Li, Yiyuan Li, Yuan-Fang Li, Zhenghua Li, Zhongli Li, Zhongyang Li, Zhoujun Li, Zichao Li, Zongxi Li, Zuchao Li, Bin Liang, Chao-Chun Liang, Chen Liang, Paul Pu Liang, Xiaobo Liang, Yunlong Liang, Lizi Liao, Jindřich Libovický, Chaya Liebeskind, Wang Lijie, Gilbert Lim, Kwan Hui Lim, Bill Yuchen Lin, Chih-Jen Lin, Chu-Cheng Lin, Hongfei Lin, Junyang Lin, Lucy H. Lin, Peiqin Lin, Ting-En Lin, Weizhe Lin, Xi Victoria Lin, Xiang Lin, Yankai Lin, Ying Lin, Zehao Lin, Zhaojiang Lin, Zheng Lin, Zhouhan Lin, Zi Lin, Zhen-Hua Ling, Tal Linzen, Pierre Lison, Johann-Mattis List, Robert Litschko, Patrick William Littell, Marina Litvak, Bin Liu, Bing Liu, Chen Cecilia Liu, Chi-Liang Liu, Dairui Liu, Danni Liu, Dexi Liu, Fangyu Liu, Fei Liu, Feifan Liu, Han Liu, Haochen Liu, Haokun Liu, Haoyan Liu, Hui Liu, Jiachang Liu, Jian Liu, Jiangming Liu, Jing Liu, Junhao Liu, Kang Liu, Lemao Liu, Ling Liu, Linqing Liu, Liyuan Liu, Maofu Liu, Ming Liu, Nelson F. Liu, Peng Liu, Qian Liu, Qianchu Liu, Shujie Liu, Shulin Liu, Siyang Liu, Tianyu Liu, Weijie Liu, Xianggen Liu, Xiao Liu, Xiao Liu, Ximeng Liu, Xin Liu, Xinchen Liu, Xudong Liu, Xuebo Liu, Xueqing Liu, Yang Janet Liu, Yijia Liu, Yijin Liu, Yingchi Liu, Yong Liu, Zemin Liu, Zhengyuan Liu, Zhiyuan Liu, Zhun Liu, Zihan Liu, Zoey Liu, Farhana Ferdousi Liza, Nikola Ljubešić, Kyle Lo, Guodong Long, Yunfei Long, José David Lopes, Lucelene Lopes, Natalia Loukachevitch, Ismini Lourentzou, Sharid Loáiciga, Di Lu, Wei Lu, Weiming Lu, Yao Lu, Yaojie Lu, Yichao Lu, Nurul Lubis, Stephanie M. Lukin, Hongyin Luo, Huaishao Luo, Ling Luo, Ping Luo, Renqian Luo, Ruotian Luo, Tianyi Luo, Anh Tuan Luu, Kelvin Luu, Shangwen Lv, Xin Lv, Teresa Lynn, Chenyang Lyu, Michael Lyu, Samuel Läubli

Ji Ma, Jianqiang Ma, Kaixin Ma, Mingyu Derek Ma, Qianli Ma, Xiaofei Ma, Xinyin Ma, Xuezhe Ma, Xutai Ma, Sean MacAvaney, Aman Madaan, Avinash Madasu, Mounica Maddela, Pranava Madhyastha, Andrea Madotto, Walid Magdy, Manuel Mager, Suchismit Mahapatra, Adyasha Maharana, Debanjan Mahata, Rahmad Mahendra, Ayush Maheshwari, Gaurav Maheshwari, Kyle Mahowald, Wolfgang Maier, Jean Maillard, Olga Majewska, Bodhisattwa Prasad Majumder, Márton Makrai, Prodromos Malakasiotis, Chaitanya Malaviya, Andreas Maletti, Ankur Mali, Eric Malmi, Christopher Malon, Radhika Mamidi, Saab Mansour, Ramesh Manuvinakurike, Emaad Manzoor, Jiaxin Mao, Wenji Mao, Xian-Ling Mao, Xin Mao, Yuren Mao, Vladislav Maraev, Ana Marasovic, Diego Marcheggiani, Daniel Marcu, Piotr Mardziel, Andreas Marfurt, Katerina Margatina, Benjamin Marie, Zita Marinho, Antonis Maronikolakis, Edison Marrese-Taylor, Hector Martinez Alonso, Pedro Henrique Martins, Yuval Marton, Sameen Maruf, Claudia Marzi, Sandeep Mathias, Prashant Mathur, Puneet Mathur, David Martins De Matos, Sérgio Matos, Yuichiroh Matsubayashi, Takuya Matsuzaki, Yevgen Matusevych, Evgeny Matusov, Kaushal Kumar Maurya, Nickil Maveli, Jonathan May, Stephen Mayhew, Karen Mazidi, Sahisnu Mazumder, Arya D. McCarthy, John Philip McCrae, Matthew B.a. McDermott, Denis Jered McInerney, Alexander Mehler, Shikib Mehri, Nikhil Mehta, Hongyuan Mei, Hardik Meisheri, Clara Isabel Meister, Dheeraj Mekala, Telmo Menezes, Fandong Meng, Rui Meng, Tao Meng, Yu Meng, Yuanliang Meng, Zaiqiao Meng, Zhao Meng, Rakesh R Menon, Samuel Mensah, Wolfgang Menzel, Paola Merlo, William Merrill, Mohsen Mesgar, Florian Metze, Donald Metzler, Marie-Jean Meurs, Haitao Mi, Yisong Miao, Julian Michael, Paul Michel, Lesly Miculicich, Sabrina J Mielke, Margot Mieskes, Todor Mihaylov, Tsvetomila Mihaylova, Elena Mikhalkova, Simon Mille, Timothy A Miller, Tristan Miller, Eleni Miltsakaki, David Mimno, Bonan Min, Sewon Min, Pasquale Minervini, Xu Mingzhou, Hideya Mino, Shachar Mirkin, Seyedabolghasem Mirroshandel, Paramita Mirza, Abhijit Mishra, Swaroop Mishra, Kanishka Misra, Masato Mita, Prasenjit Mitra, Jelena Mitrović, Arpit Mittal, Vibhu O. Mittal, Makoto Miwa, Yusuke Miyao, Takashi Miyazaki, Daichi Mochihashi, Ashutosh Modi, Hans Moen, Aditya Mogadala, Nikita Moghe, Alireza Mohammadshahi, Muqeeth Mohammed, Hosein Mohebbi, Diego Molla, Natawut Monaikul, Nicholas Monath, Ishani Mondal, Joel Ruben Antony Moniz, Syrielle Montariol, Manuel Montes, Seungwhan Moon, Ray Mooney, Nafise Sadat Moosavi, Mehrad Moradshahi, Vlad I Morariu, Erwan Moreau, Jose G Moreno, Mathieu Morey, Gaku Morio, Makoto Morishita, John Xavier Morris, David R Mortensen, Ahmadreza Mosallanezhad, Marius Mosbach, Lili Mou, Xiangyang Mou, Seyed Mahed Mousavi, Maximilian Mozes, Yassine Mrabet, Frank Martin Mtumbuka, Hamdy Mubarak, Pramod Kaushik Mudrakarta, Aaron Mueller, David Mueller, Matteo Muffo, Animesh Mukherjee, Phoebe Mulcaire, Matthew Mulholland, Deepak Muralidharan, Masayasu Muraoka, Elena Musi, Sheshera Mysore, Mark-Christoph Müller, Mathias Müller, Thomas Müller

Seung-Hoon Na, Nona Naderi, Masaaki Nagata, Ajay Nagesh, Saeed Najafi, Tetsuji Nakagawa, Diane Napolitano, Jason Naradowsky, Karthik R Narasimhan, Tahira Naseem, Sudip Kumar Naskar, Alexis Nasr, Vivi Nastase, Anandhavelu Natarajan, Tristan Naumann, Roberto Navigli, Matteo Negri, Graham Neubig, Günter Neumann, Mariana Neves, Denis Newman-Griffis, Dai Quoc Nguyen, Hoang Van Nguyen, Huyen Nguyen, Thanh V Nguyen, Thanh-Tung Nguyen, Thien Huu Nguyen, Truc-Vien T. Nguyen, Hoang-Quoc Nguyen-Son, Jianmo Ni, Garrett Nicolai, Massimo

Nicosia, Vlad Niculae, Feng Nie, Yixin Nie, Jan Niehues, Christina Niklaus, Fedor Nikolaev, Giannis Nikolentzos, Vassilina Nikoulina, Qiang Ning, Takashi Ninomiya, Nobal B. Niraula, Kosuke Nishida, Kyosuke Nishida, Noriki Nishida, Masaaki Nishino, Sergiu Nisioi, Guanglin Niu, Tong Niu, Xing Niu, Hiroshi Noji, Tadashi Nomoto, Damien Nouvel, Michal Novák, Pierre Nugues, Claire Nédellec, Aurélie Névéol

Alexander O'Connor, Yusuke Oda, Stephan Oepen, Maciej Ogrodniczuk, Barlas Oguz, Alice Oh, Yoo Rhee Oh, Kiyonori Ohtake, Naoaki Okazaki, Tsuyoshi Okita, Manabu Okumura, Hugo Gonçalo Oliveira, Antoni Oliver, Arturo Oncevay, Yasumasa Onoe, Juri Opitz, Shereen Oraby, John Ortega, Pedro Ortiz Suarez, Yohei Oseki, Malte Ostendorff, Naoki Otani, Myle Ott, Zhijian Ou, Zijing Ou, Hiroki Ouchi, Nedjma Ousidhoum, Robert Östling, Lilja Øvrelid

Maria Leonor Pacheco, Inkit Padhi, Aishwarya Padmakumar, Santanu Pal, Sukomal Pal, Chester Palen-Michel, Alexis Palmer, Endang Wahyu Pamungkas, Boyuan Pan, Liangming Pan, Liang Pang, Richard Yuanzhe Pang, Sheena Panthaplackel, Alexandros Papangelis, Nikolaos Pappas, Emerson Cabrera Paraiso, Letitia Parcalabescu, Natalie Parde, Antonio Pareja-Lora, Cecile Paris, ChaeHun Park, Chanjun Park, Hyunji Hayley Park, Jungsoo Park, Kunwoo Park, Lucy Park, Youngja Park, Ioannis Partalas, Niko Tapio Partanen, Prasanna Parthasarathi, Md Rizwan Parvez, Gabriella Pasi, Tommaso Pasini, Ramakanth Pasunuru, Or Patashnik, Arkil Patel, Kevin Patel, Raj Patel, Roma Patel, Sangameshwar Patil, Barun Patra, Braja Patra, Jasabanta Patro, Manasi Patwardhan, Siddharth Patwardhan, Debjit Paul, Silviu Paun, John Pavlopoulos, Pavel Pecina, Jiaxin Pei, Stephan Peitz, Viktor Pekar, Baolin Peng, Hao Peng, Haoruo Peng, Siyao Peng, Wei Peng, Xi Peng, Xutan Peng, Yifan Peng, Lis Pereira, Martin Pereira, Julien Perez, Gabriele Pergola, Jan-Thorsten Peter, Ben Peters, Matthew E Peters, Pavel Petrushkov, Sandro Pezzelle, Jonas Pfeiffer, Minh-Quang Pham, Quan Pham, Van-Thuy Phi, Maciej Piasecki, Massimo Piccardi, Karl Pichotta, Mohammad Taher Pilehvar, Tiago Pimentel, Aidan Pine, Juan Pino, Yuval Pinter, Flammie A Pirinen, Benjamin Piwowarski, Lonneke Van Der Plas, Bryan A. Plummer, Brian Plüss, Sylvain Pogodalla, Martin Popel, Octavian Popescu, Andrei Popescu-Belis, Fred Popowich, François Portet, Matt Post, Martin Potthast, Christopher Potts, Amir Pouran Ben Veyseh, Sandhya Prabhakaran, Vinodkumar Prabhakaran, Shrimai Prabhumoye, Aniket Pramanick, Jakob Prange, Animesh Prasad, Archiki Prasad, Judita Preiss, Audi Primadhanty, Victor Prokhorov, Prokopis Prokopidis, Haritz Puerto, Rajkumar Pujari, Matthew Purver, Valentina Pyatkin, Juan Antonio Pérez-Ortiz

Fanchao Qi, Jianzhong Qi, Peng Qi, Tao Qi, Dong Qian, Kun Qian, Yujie Qian, Libo Qin, Yujia Qin, Liang Qiu, Long Qiu, Xipeng Qiu, Chen Qu, Lizhen Qu, Xiaoye Qu

Ella Rabinovich, Gorjan Radevski, Alessandro Raganato, Dinesh Raghu, Vipul Raheja, Afshin Rahimi, Hossein Rajaby Faghihi, Sara Rajaee, Dheeraj Rajagopal, Sanguthevar Rajasekaran, Pavithra Rajendran, Geetanjali Rakshit, Dhananjay Ram, Ori Ram, Taraka Rama, Deepak Ramachandran, Anil Ramakrishna, Ganesh Ramakrishnan, Owen Rambow, Alan Ramponi, Gabriela Ramírez De La Rosa, Tharindu Ranasinghe, Surangika Ranathunga, Priya Rani, Peter A. Rankel, Jinfeng Rao, Yanghui Rao, Ahmad Rashid, Hannah Rashkin, Abhinav Rastogi, Vipul Kumar Rathore, Vikas Raunak, Shauli Ravfogel, Abhilasha Ravichander, Vinit Ravishankar, Anirudh Ravula, Avik Ray, Soumya Ray, Manny Rayner, Julia Rayz, Traian Rebedea, Sravana Reddy, Hanumant Harichandra Redkar, Georg Rehm, Marek Rei, Nils Reimers, Navid Rekabsaz, Da Ren, Feiliang Ren, Feiliang Ren, Pengjie Ren, Ruiyang Ren, Shuhuai Ren, Shuo Ren, Xiang Ren, Xuancheng Ren, Zhaochun Ren, Adi Renduchintala, Mehdi Rezagholizadeh, Saed Rezayi, Leonardo F. R. Ribeiro, Caitlin Laura Richter, Sebastian Riedel, Stefan Riezler, German Rigau, Shruti Rijhwani, Matīss Rikters, Darcey Riley, Laura Rimell, Eric Ringger, Annette Rios, Anthony Rios, Miguel Rios, Brian Roark, Kirk Roberts, Christophe Rodrigues, Pedro Rodriguez, Melissa Roemmele, Lina Maria Rojas-Barahona, Roland Roller, Stephen Roller, Alexey Romanov, Salvatore Romeo, Srikanth Ronanki, Subendhu Rongali, Rudolf Rosa, Aiala Rosá, Michael Roth, Sascha Rothe, Salim Roukos, Dmitri Roussinov, Bryan R. Routledge, Aurko Roy, Subhro Roy, Jos Rozen, Alla Rozovskaya, Dongyu Ru, Raphael Rubino, Sebastian Ruder, Koustav Rudra, Frank Rudzicz, Federico Ruggeri, Thomas Ruprecht, Alexander M Rush, Irene Russo, Phillip Rust, Attapol Rutherford, Max Ryabinin, Maria Ryskina, Andreas Rücklé

C S, Ashish Sabharwal, Mrinmaya Sachan, Fatiha Sadat, Arka Sadhu, Marzieh Saeidi, Niloofar Safi Samghabadi, Kenji Sagae, Horacio Saggion, Monjoy Saha, Swarnadeep Saha, Tulika Saha, Saurav Sahay, Gaurav Sahu, Sunil Kumar Sahu, Hassan Sajjad, Keisuke Sakaguchi, Sakriani Sakti, Elizabeth Salesky, Alexandre Salle, Avneesh Saluja, Tanja Samardzic, Younes Samih, Danae Sanchez Villegas, Chinnadhurai Sankar, Malaikannan Sankarasubbu, Sashank Santhanam, Marina Santini, Bishal Santra, Sebastin Santy, Maarten Sap, Naomi Saphra, Maya Sappelli, Zahra Sarabi, Sheikh Muhammad Sarwar, Felix Sasaki, Shota Sasaki, Ryohei Sasano, Giorgio Satta, Danielle Saunders, Agata Savary, Aleksandar Savkov, Beatrice Savoldi, Apoorv Umang Saxena, Asad B. Sayeed, Thomas Schaaf, Shigehiko Schamoni, Tatjana Scheffler, Christian Scheible, Yves Scherrer, Timo Schick, Marten Van Schijndel, Frank Schilder, Viktor Schlegel, Jonathan Schler, Helmut Schmid, Tyler Schnoebelen, Steven Schockaert, Alexandra Schofield, Sabine Schulte Im Walde, Claudia Schulz, Hannes Schulz, Elliot Schumacher, Anne-Kathrin Schumann, Sebastian Schuster, Tal Schuster, Roy Schwartz, Robert Schwarzenberg, Stefan Schweter, Johannes Schäfer, Djamé Seddah, João Sedoc, Satoshi Sekine, David Semedo, Nasredine Semmar, Sina Semnani, Lütfi Kerem Senel, Rico Sennrich, Minjoon Seo, Yeon Seonwoo, Christophe Servan, Lei Sha, Izhak Shafran, Darsh Jaidip Shah, Kashif Shah, Samira Shaikh, Cory Shain, Chao Shang, Guokan Shang, Jingbo Shang, Mingyue Shang, Chenze Shao, Nan Shao, Yutong Shao, Zhihong Shao, Ori Shapira, Naomi Tachikawa Shapiro, Amr Sharaf, Arpit Sharma, Ashish Sharma, Vasu Sharma, Serge Sharoff, Rebecca Sharp, Hassan Shavarani, Peter Shaw, Qiaoqiao She, Zaid Sheikh, Artem Shelmanov, Hua Shen, Jiaming Shen, Lei Shen, Qinlan Shen, Sheng Shen, Shiqi Shen, Tao Shen, Xiaoyu Shen, Yikang Shen, Yilin Shen, Yongliang Shen, Emily Sheng, Qiang Sheng, Tom Sherborne, Chuan Shi, Freda Shi, Jiatong Shi, Jiaxin Shi, Ning Shi, Peng Shi, Shuming Shi, Tianze Shi, Weijia Shi, Weiyan Shi, Xing Shi, Yangyang Shi, Zhouxing Shi, Tomohide Shibata, Nobuyuki Shimizu, Anastasia Shimorina, Jamin Shin, Yow-Ting Shiue, Boaz Shmueli, Eyal Shnarch, Linjun Shou, Mohit Shridhar, Akshat Shrivastava, Manish Shrivastava, Kai Shu, Lei Shu, Raphael Shu, Kurt Shuster, Vered Shwartz, Chenglei Si, Mei Si, Aditya Siddhant, A.b. Siddique, Carina Silberer, Miikka Silfverberg, Khalil Sima'an, Patrick Simianer, Kathleen Siminyu, Arabella Jane Sinclair, Sameer Singh, Karan Singla, Koustuv Sinha, Kairit Sirts, Amy Siu, Milena Slavcheva, Noam Slonim, David A. Smith, Felipe Soares, Christine Soh, Haoyu Song, Hyun-Je Song, Kai Song, Kaiqiang Song, Linfeng Song, Mingyang Song, Ruihua Song, Wei Song, Xingyi Song, Yiping Song, Sandeep Soni, Rishi Sonthalia, Claudia Soria, Alexey Sorokin, Daniil Sorokin, William Eduardo Soto Martinez, Sajad Sotudeh, Marlo Souza, Lucia Specia, Matthias Sperber, Vivek Srikumar, Balaji Vasan Srinivasan, Tejas Srinivasan, Shashank Srivastava, Edward P. Stabler, Felix Stahlberg, Ieva Staliunaite, Marija Stanojevic, Gabriel Stanovsky, David Stap, Katherine Stasaski, Manfred Stede, Mark Steedman, Benno Stein, Shane Steinert-Threlkeld, Elias Stengel-Eskin, Amanda Stent, Mark Stevenson, Ian Stewart, Matthew Stone, Kevin Stowe, Karl Stratos, Kristina Striegnitz, Heiner Stuckenschmidt, Nikolaos Stylianou, Sara Stymne, Dan Su, Hui Su, Jinsong Su, Keh-Yih Su, Shang-Yu Su, Weifeng Su, Yu Su, Yusheng Su, Nishant Subramani, Lakshmi Subramanian, Sanjay Subramanian, Katsuhito Sudoh, Saku Sugawara, Hiroaki Sugiyama, Alessandro Suglia, Yoshihiko Suhara, Dianbo Sui, Zhifang Sui, Elior Sulem, Md Arafat Sultan, Changzhi Sun, Chengjie Sun, Fei Sun, Haipeng Sun, Haitian Sun, Huan Sun, Jian Sun, Jingyi Sun, Kai Sun, Kai Sun, Ming Sun, Mingming Sun, Si Sun, Simeng Sun, Siqi Sun, Tianxiang Sun, Yawei Sun, Yibo Sun, Yifan Sun, Yu Sun, Zequn Sun, Zhiqing Sun, Dhanasekar Sundararaman, Mujeen Sung, Hanna Suominen, Mihai Surdeanu, Anshuman Suri, Shiv Surya, Simon Suster, Mirac Suzgun, Jun Suzuki, Masatoshi Suzuki, Swabha Swayamdipta, Benjamin Sznajder, Stan Szpakowicz, Felipe

Sánchez-Martínez, Gözde Gül Şahin

Ryuki Tachibana, Oyvind Tafjord, Shabnam Tafreshi, Hiroya Takamura, Ryuichi Takanobu, Sho Takase, Ece Takmaz, Aarne Talman, Derek Tam, George Tambouratzis, Fabio Tamburini, Akihiro Tamura, Chuanqi Tan, Fei Tan, Liling Tan, Samson Tan, Xu Tan, Zeqi Tan, Kumiko Tanaka-Ishii, Buzhou Tang, Gongbo Tang, Hao Tang, Qingming Tang, Raphael Tang, Shuai Tang, Siliang Tang, Yi-Kun Tang, Zhiwen Tang, Ludovic Tanguy, Xavier Tannier, Chongyang Tao, Shiva Taslimipoor, Sandeep Tata, Yuka Tateisi, Michiaki Tatsubori, Marta Tatu, Hillel Taub-Tabib, Yi Tay, Andon Tchechmedjiev, Christoph Teichmann, Selma Tekir, Serra Sinem Tekiroglu, Eric S. Tellez, Irina Temnikova, Zhiyang Teng, Ian Tenney, Hiroki Teranishi, Silvia Terragni, Alberto Testoni, Nithum Thain, Khushboo Thaker, Urmish Thakker, Nandan Thakur, Kilian Theil, Jesse Thomason, Laure Thompson, Sam Thomson, Camilo Thorne, James Thorne, Junfeng Tian, Ran Tian, Yingtao Tian, Zhiliang Tian, Jörg Tiedemann, Tiago Timponi Torrent, Erik Tjong Kim Sang, Gaurav Singh Tomar, Nadi Tomeh, Nicholas Tomlin, Sara Tonelli, Mariya Toneva, MeiHan Tong, Antonio Toral, Kentaro Torisawa, Samia Touileb, Julien Tourille, Quan Hung Tran, Dietrich Trautmann, Marcos Vinicius Treviso, Hai-Long Trieu, Alina Trifan, Enrica Troiano, Tuan Quoc Truong, Chen-Tse Tsai, Bo-Hsiang Tseng, Masaaki Tsuchida, Yoshimasa Tsuruoka, Kewei Tu, Lifu Tu, Mei Tu, Zhaopeng Tu, Iulia Raluca Turc, Martin Tutek, Francis M. Tyers, Andre Tättar

Rutuja Ubale, Ana Sabina Uban, Takuma Udagawa, Umair Ul Hassan, Stefan Ultes, Shyam Upadhyay, L. Alfonso Ureña, Ricardo Usbeck

Keyon Vafa, Sowmya Vajjala, Jannis Vamvas, Tim Van De Cruys, Benjamin Van Durme, Emiel Van Miltenburg, Rik Van Noord, Keith N VanderLinden, Lucy Vanderwende, David Vandyke, Natalia Vanetik, Daniel Varab, Siddharth Varia, Lucy Vasserman, Julien Velcin, Alakananda Vempala, Sriram Venkatapathy, Giulia Venturi, Suzan Verberne, Gaurav Verma, Rakesh M Verma, Giorgos Vernikos, Yannick Versley, Karin Verspoor, Anvesh Rao Vijjini, David Vilar, Jesús Vilares, Serena Villata, Aline Villavicencio, Éric Villemonte De La Clergerie, Veronika Vincze, Krishnapriya Vishnubhotla, Ngoc Phuoc An Vo, Rob Voigt, Elena Voita, Soroush Vosoughi, Thang Vu, Thuy Vu, Thuy-Trang Vu, Tu Vu, Xuan-Son Vu, Yogarshi Vyas, Ekaterina Vylomova

Henning Wachsmuth, Takashi Wada, Joachim Wagner, Byron C Wallace, Mengting Wan, Mingyu Wan, Stephen Wan, Yao Wan, Yu Wan, Alex Wang, Bailin Wang, Baoxin Wang, Baoxun Wang, Bin Wang, Bingqing Wang, Boxin Wang, Changhan Wang, Chao Wang, Chenguang Wang, Chengyu Wang, Cunxiang Wang, Daling Wang, Dingmin Wang, Fei Wang, Guangrun Wang, Guoyin Wang, Hai Wang, Han Wang, Hanrui Wang, Hao Wang, Hao Wang, Haohan Wang, Haoyu Wang, Hong Wang, Hongfei Wang, Hua Wang, Jin Wang, Jin Wang, Jingang Wang, Jingkang Wang, Jue Wang, Ke Wang, Liang Wang, Lidan Wang, Lingzhi Wang, Liwen Wang, Lucy Lu Wang, Ping Wang, Pinghui Wang, Qiang Wang, Qifan Wang, Qingyun Wang, Quan Wang, Rui Wang, Rui Wang, Runze Wang, Shaonan Wang, Shi Wang, Shuo Wang, Shuohang Wang, Sinong Wang, Tong Wang, Tong Wang, Wei Wang, Wei Wang, Weiyue Wang, Wen Wang, Wenbo Wang, Wenhui Wang, Wenya Wang, Xiaojie Wang, Xiaolin Wang, Xiaozhi Wang, Xin Wang, Xing Wang, Xinyi Wang, Xuezhi Wang, Yan Wang, Yaqing Wang, Yequan Wang, Yifei Wang, Yijue Wang, Yile Wang, Yingyao Wang, Yiran Wang, Yizhong Wang, Yong Wang, Yue Wang, Yue Wang, Yujing Wang, Zhen Wang, Zhichun Wang, Zhongqing Wang, Zijian Wang, Ziqi Wang, Zirui Wang, Leo Wanner, Nigel G. Ward, Alex Warstadt, Christian Wartena, Koki Washio, Ingmar Weber, Leon Weber, Noah Weber, Kellie Webster, Julie Weeds, Jason Wei, Johnny Wei, Junqiu Wei, Penghui Wei, Wei Wei, Xiangpeng Wei, Xiaochi Wei, Shira Wein, David Weir, Ralph M. Weischedel, Charles Welch, Orion Weller, Haoyang Wen, Lijie Wen, Rongxiang Weng, Peter West, Taesun Whang, Michael White, Michael Wiegand, Sarah Wiegreffe, Adam Wiemerslage, Derry Wijaya, Gijs Wijnholds, Ethan Wilcox, Rodrigo Wilkens, Jake Ryland Williams, Jennifer Williams, Shomir Wilson, Steven R. Wilson, Genta Indra Winata, Shuly Wintner, Sam Wiseman, Guillaume Wisniewski, Magdalena Wolska, Derek F. Wong, Tak-Lam Wong, Dina Wonsever, Zach Wood-Doughty, Bo Wu, Bowen Wu, Chien-Sheng Wu, Chuhan Wu, Chun-Kai Wu, Dayong Wu, Di Wu, Fangzhao Wu, Jian Wu, Junshuang Wu, Lianwei Wu, Lijun Wu, Lingfei Wu, Qianhui Wu, Qingyang Wu, Shijie Wu, Shuangzhi Wu, Sixing Wu, Stephen Wu, Tongshuang Wu, Wei Wu, Xianchao Wu, Xiaobao Wu, Yanan Wu, Youzheng Wu, Yu Wu, Yuanbin Wu, Yunfang Wu, Yuting Wu, Zeqiu Wu, Zhen Wu, Zhiyong Wu, Zhonghai Wu, Joern Wuebker

Congying Xia, Jingbo Xia, Mengzhou Xia, Patrick Xia, Qingrong Xia, Rui Xia, Yikun Xian, Jiannan Xiang, Rong Xiang, Chaojun Xiao, Chunyang Xiao, Huiru Xiao, Jinghui Xiao, Lin Xiao, Liqiang Xiao, Min Xiao, Tong Xiao, Wen Xiao, Yanghua Xiao, Boyi Xie, Jun Xie, Qianqian Xie, Ruobing Xie, Tianbao Xie, Yuqiang Xie, Ji Xin, Frank Xing, Deyi Xiong, Wenhan Xiong, Benfeng Xu, Boyan Xu, Can Xu, Canwen Xu, Chen Xu, Dongkuan Xu, Frank F. Xu, Hongfei Xu, Hu Xu, Jia Xu, Jiacheng Xu, Jinan Xu, Jing Xu, Jingjing Xu, Jitao Xu, Jun Xu, Kun Xu, Lu Xu, Peng Xu, Peng Xu, Qiantong Xu, Qiongkai Xu, Ruifeng Xu, Runxin Xu, Ruochen Xu, Shusheng Xu, Wang Xu, Weijia Xu, Weiran Xu, Weiwen Xu, Wenduan Xu, Xinnuo Xu, Yan Xu, Yang Xu, Yumo Xu, Zenglin Xu, Zhen Xu, Zhiyang Xu

Shuntaro Yada, Vikas Yadav, Yadollah Yaghoobzadeh, Ikuya Yamada, Ivan P. Yamshchikov, Hanqi Yan, Jun Yan, Lingyong Yan, Yu Yan, Yuanmeng Yan, Baosong Yang, Changbing Yang, Chenghao Yang, Fan Yang, Haiqin Yang, Jie Yang, Jun Yang, Linyi Yang, Min Yang, Mingming Yang, Muyun Yang, Ruosong Yang, Sen Yang, Sen Yang, Songlin Yang, Tsung-Yen Yang, Wei Yang, Wenmian Yang, Yilin Yang, Yinfei Yang, Yujiu Yang, Zhao Yang, Zhen Yang, Zhichao Yang, Zhilin Yang, Ziqing Yang, Ziyi Yang, Jianmin Yao, Liang Yao, Shunyu Yao, Wenlin Yao, Ziyu Yao, Mark Yatskar, Deming Ye, Qinyuan Ye, Reyyan Yeniterzi, Jinyoung Yeo, Xiaoyuan Yi, Seid Muhie Yimam, Da Yin, Pengcheng Yin, Qingyu Yin, Xuwang Yin, Yichun Yin, Sho Yokoi, Zheng Xin Yong, Kang Min Yoo, Seunghyun Yoon, Masashi Yoshikawa, Steve Young, Safoora Yousefi, Bei Yu, Bowen Yu, Changlong Yu, Chen Yu, Dian Yu, Dian Yu, Dong Yu, Heng Yu, Hong Yu, Jifan Yu, Juntao Yu, Kai Yu, Mo Yu, Tao Yu, Tiezheng Yu, Wenhao Yu, Xiaodong Yu, Yue Yu, Caixia Yuan, Jianhua Yuan, Nicholas Jing Yuan, Yu Yuan, Zheng Yuan, Xiang Yue, Hyokun Yun

Annie Zaenen, Wajdi Zaghouani, Marcos Zampieri, Marcely Zanon Boito, Alessandra Zarcone, Sina Zarrieß, Vicky Zayats, Rabih Zbib, Albin Zehe, Rowan Zellers, Yury Zemlyanskiy, Daojian Zeng, Fengzhu Zeng, Jiali Zeng, Jichuan Zeng, Qi Zeng, Shuang Zeng, Weixin Zeng, Xingshan Zeng, Zhiyuan Zeng, Thomas Zenkel, Deniz Zeyrek, Hanwen Zha, Fangzhou Zhai, Haolan Zhan, Li-Ming Zhan, Runzhe Zhan, Biao Zhang, Bowen Zhang, Bowen Zhang, Chen Zhang, Chen Zhang, Chiyu Zhang, Chuheng Zhang, Danqing Zhang, Dawei Zhang, Delvin Ce Zhang, Denghui Zhang, Dong Zhang, Dongdong Zhang, Dongxu Zhang, Dongyu Zhang, Guanhua Zhang, Haibo Zhang, Hainan Zhang, Haisong Zhang, Hao Zhang, Hao Zhang, Haoyu Zhang, Hongming Zhang, Hu Zhang, Jiajun Zhang, Jianguo Zhang, Jieyu Zhang, Jinchao Zhang, Jingqing Zhang, Ke Zhang, Kun Zhang, Lei Zhang, Lei Zhang, Li Zhang, Licheng Zhang, Longyin Zhang, Meishan Zhang, Meng Zhang, Michael JQ Zhang, Mike Zhang, Min Zhang, Ningyu Zhang, Peng Zhang, Qi Zhang, Richong Zhang, Rui Zhang, Ruixiang Zhang, Sheng Zhang, Shiyue Zhang, Shujian Zhang, Shuo Zhang, Tianlin Zhang, Tong Zhang, Tongtao Zhang, Wei Zhang, Wei Emma Zhang, Weinan Zhang, Wen Zhang, Wen Zhang, Xiang Zhang, Xiao Zhang, Xiaotong Zhang, Xingxing Zhang, Xinliang Frederick Zhang, Xinsong Zhang, Xinyuan Zhang, Xuanwei Zhang, Xuanyu Zhang, Xuchao Zhang, Yan Zhang, Yan Zhang, Yao Zhang, Yichi Zhang, Yu Zhang, Yu Zhang, Yuan Zhang, Yuanzhe Zhang, Yue Zhang, Yuhao Zhang, Yuhui Zhang, Yunyi Zhang, Yusen Zhang, Zeyu Zhang, Zheng Zhang, Zhengyan Zhang, Zhihao Zhang, Zhirui Zhang, Zhisong Zhang, Zhuosheng Zhang, Ziqi Zhang, Chao Zhao, Chen Zhao, Dongyan Zhao, Guangxiang Zhao, Jieyu Zhao, Kai Zhao, Mengjie Zhao, Sanqiang Zhao, Tiancheng Zhao, Tianyu Zhao, Tiejun Zhao, Yang Zhao, Yanpeng

Zhao, Yao Zhao, Yilun Zhao, Zhenjie Zhao, Zhou Zhao, Bo Zheng, Changmeng Zheng, Chujie Zheng, Renjie Zheng, Xiaoqing Zheng, Yinhe Zheng, Zaixiang Zheng, Ming Zhong, Peixiang Zhong, Victor Zhong, Wanjun Zhong, Zexuan Zhong, Ben Zhou, Chunting Zhou, Deyu Zhou, Dong Zhou, Giulio Zhou, Guangyou Zhou, Jiawei Zhou, Jie Zhou, Jie Zhou, Jingbo Zhou, Junpei Zhou, Junsheng Zhou, Li Zhou, Long Zhou, Meng Zhou, Pei Zhou, Qingyu Zhou, Shuyan Zhou, Wangchunshu Zhou, Wenxuan Zhou, Xiang Zhou, Xiangyang Zhou, Yaqian Zhou, Yi Zhou, Yichao Zhou, Yichu Zhou, Yilun Zhou, Yucheng Zhou, Zhengyu Zhou, Zhihan Zhou, Conghui Zhu, Hao Zhu, Jian Zhu, Jun Zhu, Junnan Zhu, Kenny Q. Zhu, Lixing Zhu, Muhua Zhu, Qi Zhu, Qingfu Zhu, Qinglin Zhu, Su Zhu, Wei Zhu, Xiaoyan Zhu, Yilun Zhu, Yong Zhu, Zining Zhu, Fuzhen Zhuang, Yimeng Zhuang, Caleb Ziems, Roger Zimmermann, Heike Zinsmeister, Ayah Zirikly, Shi Zong, Bowei Zou, Yanyan Zou, Amal Zouaq, Arkaitz Zubiaga, Pierre Zweigenbaum

Outstanding Action Editors

Antonios Anastasopoulos, David Bamman, Steven Bethard, Leonid Boytsov, Paula Carvalho, Snigdha Chaturvedi, Raj Dabre, Daniel Dakota, Johannes Daxenberger, Leon Derczynski, Greg Durrett, Michael Elhadad, Allyson Ettinger, Goran Glavaš, David Harwath, Shubhra Kanti Karmaker, Daniel Khashabi, Mamoru Komachi, Carolin Lawrence, John Lawrence, Constantine Lignos, Saif M. Mohammad, Philippe Muller, Rebecca J. Passonneau, Emily Prud'hommeaux, Mrinmaya Sachan, Lane Schwartz, Kevin Small, Efstathios Stamatatos, Amanda Stent, Amalia Todirascu, Junichi Tsujii, Suzan Verberne, Antonio Jimeno Yepes, François Yvon, Luke Zettlemoyer, Justine Zhang

Outstanding Reviewers

Nader Akoury, Gianni Barlacchi, Rachel Bawden, Gábor Bella, Delphine Bernhard, Shruti Bhosale, Michael Bloodgood, Ondrej Bojar, Iacer Calixto, Rémi Cardon, Thiago Castro Ferreira, Tuhin Chakrabarty, Verna Dankers, Yupei Du, Micha Elsner, Antske Fokkens, Stella Frank, Alexander Fraser, Dayne Freitag, Daniel Fried, Dan Garrette, Philip John Gorinski, Dagmar Gromann, Liane Guillou, Jack Hessel, Nanjiang Jiang, Gareth J. F. Jones, Min-Yen Kan, Anna Kazantseva, Fajri Koto, Julia Kreutzer, Kalpesh Krishna, Dawn Lawrie, Andrew Lee, Jordan Lee Boyd-Graber, Gina-Anne Levow, Xiang Lisa Li, Patrick William Littell, Kaixin Ma, Vladislav Maraev, Alexander Mehler, Florian Metze, Julian Michael, Paul Michel, Elena Musi, Sheshera Mysore, Denis Newman-Griffis, Tong Niu, Michal Novák, Siddharth Patwardhan, Karl Pichotta, Yuval Pinter, Peng Qi, Surangika Ranathunga, Vikas Raunak, Pedro Rodriguez, Sebastian Ruder, Alexander M. Rush, Elizabeth Salesky, Thomas Schaaf, Yves Scherrer, Viktor Schlegel, Elliot Schumacher, Ian Stewart, Naomi Tachikawa Shapiro, Emiel van Miltenburg, Peter West, Adam Wiemerslage, Jitao Xu, Yue Yu, Yury Zemlyanskiy

Keynote Talk: Language in the Human Brain

Angela D. Friederici
Max Planck Institute for Human Cognitive and Brain Sciences, Leipzig, Germany

Abstract: Language is considered to be a uniquely human faculty. The different aspects of the language system, namely phonology, semantics and syntax have long been discussed with respect to their species-specificity. Syntax as the ability to process hierarchical structures appears to be specific to humans. The available neuroscientific data allow us to define the functional language network which involves Broca's area in the inferior frontal cortex and the posterior superior temporal cortex. Within this network, the posterior part of Broca's area plays a special role as it supports the processing of hierarchical syntactic structures, in particular the linguistic computation Merge which is at the root of every language. This part of Broca's area is connected to the posterior temporal cortex via a dorsally located white matter fiber tract hereby providing to structural basis for the functional interplay of these regions. It has been shown that the maturation of this white matter pathway is directly correlated with the ability to process syntactically complex sentences during human development. Moreover, this dorsal pathway appears to be weak in the prelinguistic infant and in the non-human primate. These findings suggest that the dorsal pathway plays a crucial role in the emergence of syntax in human language.

Bio: Angela D. Friederici is a cognitive neuroscientist in the domain of language. She is director at the Max Planck Institute for Human Cognitive and Brain Sciences (MPI CBS) in Leipzig, Germany and the Founding director of this institution founded in 1994.
She graduated in linguistics and psychology at the University of Bonn (Germany) and spent a postdoctoral year at MIT (USA). She was a research fellow at the Max Planck Institute in Nijmegen (NL), at the University Rene Descartes, Paris (F) and University of California, San Diego (USA). Prior to joining the Max Planck Society as a director, she was professor for Cognitive Sciences at the Free University Berlin. Friederici is honorary professor at the University of Leipzig (Psychology), the University of Potsdam (Linguistics) and the Charité Universitätsmedizin Berlin (Neurology) and she holds a Doctor honoris causa from the University of Mons, Belgium. Between 2014 and 2020 she was Vice President for the Human Sciences Section of the Max Planck Society.
Her main field of research is the neurobiology of language. She published about 500 scientific papers on this topic in major international journals. She received a number of scientific awards: 1987 Heisenberg Fellowship of the German Research Foundation, 1990 Alfried Krupp Award of the Alfried Krupp von Bohlen and Halbach-Stiftung, 1997 Gottfried Wilhelm Leibniz Prize of the German Research Foundation, and 2011 Carl Friedrich Gauss Medal of the Brunswick Scientific Society. She is member of the Berlin-Brandenburg Academy of Sciences and Humanities, member of the national German Academy of Sciences 'Leopoldina' and member of the Academia Europaea.

Keynote Fire-Side Chat with Barbara Grosz and Yejin Choi on “The Trajectory of ACL and the Next 60 Years”

For the 60th Anniversary of ACL 2022, we will feature a keynote fire-side chat on “***The Trajectory of ACL and the Next 60 years***” with two keynote talks in dialogue: Barbara Grosz and Yejin Choi followed by a moderated discussion lead by Rada Mihalcea.

Remarks on What the Past Can Tell the Future

Barbara J. Grosz
Harvard University SEAS

Abstract: Research in computational linguistics and spoken language systems has made astonishing progress in the last decade. Even so, the challenge remains of achieving human-level fluent dialogue conversational capabilities beyond narrowly defined domains and tasks. Findings of earlier ACL times research on dialogue hold some lessons for breaking the “dialogue boundary” in computational linguistics yet again, if ways can be found to integrate them into deep-learning language models. These models raise some of the most serious ethical challenges of current computing research and technologies. Expanding their powers in this direction will raise more. In discussing these topics, I will raise questions for Prof. Choi and our subsequent discussion.

Bio: Barbara J. Grosz is Higgins Research Professor of Natural Sciences in the Paulson School of Engineering and Applied Sciences at Harvard University. Her contributions to AI include fundamental advances in natural-language dialogue processing and in theories of multi-agent collaboration as well as innovative uses of models developed in this research to improve healthcare coordination and science education. She co-founded Harvard’s Embedded EthiCS program, which integrates teaching of ethical reasoning into core computer science courses. A member of the National Academy of Engineering, the American Philosophical Society, and the American Academy of Arts and Sciences, she is a fellow of several scientific societies and recipient of the 2009 ACM/AAAI Allen Newell Award, the 2015 IJCAI Award for Research Excellence, and the 2017 Association for Computational Linguistics Lifetime Achievement Award.

2082: An ACL Odyssey
The Dark Matter of Intelligence and Language

Yejin Choi
Paul G. Allen School of Computer Science & Engineering at the University of Washington

Abstract: In this talk, I will wander around reflections on the past of ACL and speculations on the future of ACL. This talk will be purposefully imaginative and accidentally controversial, by emphasizing on the importance of deciphering the dark matter of intelligence, by arguing for embracing all the ambiguous aspects of language at all pipelines of language processing, by highlighting the counterintuitive continuum across language, knowledge, and reasoning, and by pitching the renewed importance of formalisms, algorithms, and structural inferences in the modern deep learning era. Looking back, at the 50’th ACL, I couldn’t possibly imagine that I would be one day giving this very talk. For that reason, I will also share my personal anecdotes on the lasting inspirations from the previous lifetime achievement award speeches, how I believe talent is made, not born, and the implication of that belief for promoting diversity and equity.

Bio: Yejin Choi is Brett Helsel Professor at the Paul G. Allen School of Computer Science & Engineering at the University of Washington and a senior research manager at AI2 overseeing the project Mosaic. Her research investigates commonsense knowledge and reasoning, neuro-symbolic integration, neural language generation and degeneration, multimodal representation learning, and AI for social good. She is a co-recipient of the ACL Test of Time award in 2021, the CVPR Longuet-Higgins Prize in 2021, a NeurIPS Outstanding Paper Award in 2021, the AAAI Outstanding Paper Award in 2020, the Borg Early Career Award in 2018, the inaugural Alexa Prize Challenge in 2017, IEEE AI's 10 to Watch in 2016, and the ICCV Marr Prize in 2013.

Keynote Panel: Supporting Linguistic Diversity

Chair: **Steven Bird, Charles Darwin University**

Panelists and languages represented:

- Robert Jimerson, Rochester Institute of Technology (Seneca, USA)
- Fajri Koto, The University of Melbourne (Minangkabau, Indonesia)
- Heather Lent, University of Copenhagen (Creole languages)
- Teresa Lynn, Dublin City University (Irish)
- Manuel Mager, University of Stuttgart (Wixaritari, Mexico)
- Perez Ogayo, Carnegie Mellon University (Luo and Kiswahili, Kenya)

How do the tools and techniques of computational linguistics serve the full diversity of the world's languages? In particular, how do they serve the people who are still speaking thousands of local languages, often in highly multilingual, post-colonial situations? This 60th meeting of the ACL features a special theme track on language diversity with the goal of "reflecting and stimulating discussion about how the advances in computational linguistics and natural language processing can be used for promoting language diversity". This keynote talk-panel will showcase the special theme and identify key learnings from the conference. We hope this session will help to shape the future agenda for speech and language technologies in support of global linguistic diversity. The session will be organised around a series of questions under three headings.

Diverse Contexts. What is the situation of local languages where panel members are working? Are there multiple languages with distinct functions and ideologies? What are the local aspirations for the future of these languages. How are people advocating for language technology on the ground? How did the work begin? What does success look like?

Understanding Risks. Do the people who provide language data fully understand the ways their data might be used in future, including ways that might not be in their interest? What benefit are local participants promised in return for their participation, and do they actually receive these benefits? Are there harms that come with language standardisation? What principles of doing no harm can we adopt?

New Challenges. How can we provide benefits of text technologies without assuming language standardisation, official orthography, and monolingual usage? When working with local communities, do we always require data in exchange for technologies, or is a non-extractive NLP possible? How do we decolonise speech and language technology? At the beginning of the International Decade of Indigenous Languages 2022–2032, we ask: how do we respond as a community, and how can our field be more accessible to indigenous participation?

Table of Contents

BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models

Elad Ben-Zaken[1] **Shauli Ravfogel**[1,2] **Yoav Goldberg**[1,2]
[1]Computer Science Department, Bar Ilan University
[2]Allen Institute for Artificial Intelligence
{benzakenelad, shauli.ravfogel, yoav.goldberg}@gmail.com

Abstract

We introduce BitFit, a sparse-finetuning method where only the bias-terms of the model (or a subset of them) are being modified. We show that with small-to-medium training data, applying BitFit on pre-trained BERT models is competitive with (and sometimes better than) fine-tuning the entire model. For larger data, the method is competitive with other sparse fine-tuning methods. Besides their practical utility, these findings are relevant for the question of understanding the commonly-used process of finetuning: they support the hypothesis that finetuning is mainly about exposing knowledge induced by language-modeling training, rather than learning new task-specific linguistic knowledge.

1 Introduction

Large pre-trained transformer based language models, and in particular bidirectional masked language models from the BERT family (Devlin et al., 2018; Liu et al., 2019; Joshi et al., 2019), are responsible for significant gains in many NLP tasks. Under the common paradigm, the model is pre-trained on large, annotated corpora with the LM objective, and then *finetuned* on task-specific supervised data. The large size of these models make them expensive to train and, more importantly, expensive to deploy. This, along with theoretical questions on the extent to which finetuning must change the original model, has led researchers to consider fine-tuning variants where one identifies a small subset of the model parameters which need to be changed for good performance in end-tasks, while keeping all others intact (§2).

We present a simple and effective approach to fine tuning (§3), which has the following benefits:

1. Changing very few parameters per fine-tuned task.
2. Changing the same set of parameters for every tasks (task-invariance).
3. The changed parameters are both isolated and localized across the entire parameter space.
4. For small to medium training data, changing only these parameters reaches the same task accuracy as full fine-tuning, and sometimes even improves results.

Specifically, we show that freezing most of the network and **fine-tuning only the bias-terms** is surprisingly effective. Moreover, if we allow the tasks to suffer a small degradation in performance, we can fine-tune only two bias components (the "query" and "middle-of-MLP" bias terms), amounting to half of the bias parameters in the model, and only 0.04% of all model parameters.

This result has a large practical utility in deploying multi-task fine-tuned models in memory-constrained environments, as well as opens the way to trainable hardware implementations in which most of the parameters are fixed. Additionally, it opens up a set of research directions regarding the role of bias terms in pre-trained networks, and the dynamics of the fine-tuning process.

2 Background: fine-tuning and parameter-efficient fine-tuning

In transfer-learning via model fine-tuning, a pre-trained encoder network takes the input and produces contextualized representations. Then, a task-specific classification layer (here we consider linear classifiers) is added on top of the encoder, and the entire network (encoder+task specific classifiers) is trained end-to-end to minimize the task loss.

Desired properties. While fine-tuning per-task is very effective, it also results in a unique, large model for each pre-trained task, making it hard to reason about what was changed in the fine-tuning process, as well as hard to deploy, especially as the number of tasks increases. Ideally, one would want a fine-tuning method that:
(i) matches the results of a fully fine-tuned model;

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 1 - 9
May 22-27, 2022

(ii) changes only a small portion of the model's parameters; and (iii) enables tasks to arrive in a stream, instead of requiring simultaneous access to all datasets. For efficient hardware based deployments, it is further preferred that (iv): the set of parameters that change values is consistent across different tasks.

Learning vs. Exposing. The feasibility of fulfilling the above requirements depends on a fundamental question regarding the nature of the fine-tuning process of large pre-trained LMs: to what extent does the fine-tuning process induces the *learning of new capabilities*, vs. the *exposing of existing capabilities*, which were learned during the pre-training process.

Existing approaches. Two recent works have demonstrated that adaptation to various end-tasks can in fact be achieved by changing only a small subset of parameters. The first work, by Houlsby et al. (2019) ("Adapters"), achieves this goal by injecting small, trainable task-specific "adapter" modules between the layers of the pre-trained model, where the original parameters are shared between tasks. The second work, by Guo et al. (2020) ("Diff-Pruning"), achieves the same goal by adding a sparse, task-specific difference-vector to the original parameters, which remain fixed and are shared between tasks. The difference-vector is regularized to be sparse. Both methods allow adding only a small number of trainable parameters per-task (criteria ii), and each task can be added without revisiting previous ones (criteria iii).

They also partially fulfill criteria (i), suffering only a small drop in performance compared to full fine-tuning. The Adapter method, but not the Diff-Pruning method, also supports criteria (iv). However, Diff-Pruning is more parameter efficient than the Adapter method (in particular, it adds no new parameters), and also achieves better task scores. We compare against Diff-Pruning and Adapters in the experiments section, and show that we perform favorably on many tasks while also satisfying criteria (iv).

3 Bias-terms Fine-tuning (BitFit)

We propose a method we call BitFit[1] (BIas-Term FIne-Tuning), in which we freeze most of the transformer-encoder parameters, and train only the bias-terms and the task-specific classification layer. BitFit has three key properties: (i) match the results of fully fine-tuned model. (ii) enable tasks to arrive in a stream, this way it does not require simultaneous access to all datasets. (iii) fine-tune only a small portion of the model's parameters.

The approach is parameter-efficient: each new task requires storing only the bias terms parameter vectors (which amount to less than 0.1% of the total number of parameters), and the task-specific final linear classifier layer.

Concretely, the BERT encoder is composed of L layers, where each layer ℓ starts with M self-attention heads, where a self attention head (m,ℓ) has *key*, *query* and *value* encoders, each taking the form of a linear layer:

$$\begin{aligned}\mathbf{Q}^{m,\ell}(\mathbf{x}) &= \mathbf{W}_q^{m,\ell}\mathbf{x} + \mathbf{b}_q^{m,\ell}\\ \mathbf{K}^{m,\ell}(\mathbf{x}) &= \mathbf{W}_k^{m,\ell}\mathbf{x} + \mathbf{b}_k^{m,\ell}\\ \mathbf{V}^{m,\ell}(\mathbf{x}) &= \mathbf{W}_v^{m,\ell}\mathbf{x} + \mathbf{b}_v^{m,\ell}\end{aligned}$$

Where $\mathbf{x}$ is the output of the former encoder layer (for the first encoder layer $\mathbf{x}$ is the output of the embedding layer). These are then combined using an attention mechanism that does not involve new parameters:

$$\mathbf{h}_1^\ell = att\big(\mathbf{Q}^{1,\ell}, \mathbf{K}^{1,\ell}, \mathbf{V}^{1,\ell}, .., \mathbf{Q}^{m,\ell}, \mathbf{K}^{m,\ell}, \mathbf{V}^{m,l}\big)$$

and then fed to an MLP with layer-norm (LN):

$$\begin{aligned}\mathbf{h}_2^\ell &= \text{Dropout}\big(\mathbf{W}_{m_1}^\ell \cdot \mathbf{h}_1^\ell \ + \ \mathbf{b}_{m_1}^\ell\big) && (1)\\ \mathbf{h}_3^\ell &= \mathbf{g}_{LN_1}^\ell \odot \frac{(\mathbf{h}_2^\ell + \mathbf{x}) - \mu}{\sigma} + \mathbf{b}_{LN_1}^\ell && (2)\\ \mathbf{h}_4^\ell &= \ \text{GELU}\big(\mathbf{W}_{m_2}^\ell \cdot \mathbf{h}_3^\ell \ + \ \mathbf{b}_{m_2}^\ell\big) && (3)\\ \mathbf{h}_5^\ell &= \text{Dropout}\big(\mathbf{W}_{m_3}^\ell \cdot \mathbf{h}_4^\ell \ + \ \mathbf{b}_{m_3}^\ell\big) && (4)\\ \text{out}^\ell &= \mathbf{g}_{LN_2}^\ell \odot \frac{(\mathbf{h}_5^\ell + \mathbf{h}_3^\ell) - \mu}{\sigma} + \mathbf{b}_{LN_2}^\ell && (5)\end{aligned}$$

The collection of all matrices $\mathbf{W}_{(\cdot)}^{\ell,(\cdot)}$ and vectors $\mathbf{g}_{(\cdot)}^\ell$, $\mathbf{b}_{(\cdot)}^{\ell,(\cdot)}$, indicated in blue and purple are the network's *parameters* Θ, where the subset of purple vectors $\mathbf{b}_{(\cdot)}^{\ell,(\cdot)}$ are the *bias terms*.[2]

The bias terms are additive, and correspond to a very small fraction of the network, in BERT$_{\text{BASE}}$ and BERT$_{\text{LARGE}}$ bias parameters make up 0.09% and 0.08% of the total number of parameters in each model, respectively.

We show that by freezing all the parameters $\mathbf{W}^{(\cdot)}$ and $\mathbf{g}^{(\cdot)}$ and fine-tuning only the additive

[1] Our code is publicly available at www.github.com/benzakenelad/BitFit

[2] In Appendix §A.1 we relate this notation with parameter names in HuggingFace implementation.

		%Param	QNLI	SST-2	MNLI$_m$	MNLI$_{mm}$	CoLA	MRPC	STS-B	RTE	QQP	Avg.
	Train size		105k	67k	393k	393k	8.5k	3.7k	7k	2.5k	364k	
(V)	Full-FT†	100%	**93.5**	**94.1**	**86.5**	**87.1**	**62.8**	**91.9**	89.8	71.8	**87.6**	**84.8**
(V)	Full-FT	100%	91.7±0.1	93.4±0.2	85.5±0.4	85.7±0.4	62.2±1.2	90.7±0.3	**90.0±0.4**	**71.9±1.3**	87.5±0.4	84.1
(V)	Diff-Prune†	0.5%	**93.4**	**94.2**	**86.4**	**86.9**	63.5	91.3	89.5	71.5	**86.6**	**84.6**
(V)	BitFit	0.08%	91.4±2.4	93.2±0.4	84.4±0.2	84.8±0.1	**63.6±0.7**	**91.7±0.5**	**90.3±0.1**	**73.2±3.7**	85.4±0.1	84.2
(T)	Full-FT‡	100%	91.1	94.1	86.7	**86.0**	59.6	88.9	86.6	**71.2**	71.7	81.2
(T)	Full-FT†	100%	**93.4**	**94.9**	86.7	85.9	**60.5**	**89.3**	**87.6**	70.1	**72.1**	**81.8**
(T)	Adapters‡	3.6%	90.7	94.0	84.9	85.1	59.5	89.5	**86.9**	71.5	**71.8**	81.1
(T)	Diff-Prune†	0.5%	**93.3**	94.1	**86.4**	**86.0**	**61.1**	**89.7**	86.0	70.6	71.1	**81.5**
(T)	BitFit	0.08%	92.0	**94.2**	84.5	84.8	59.7	88.9	85.5	**72.0**	70.5	80.9

Table 1: BERT$_{\text{LARGE}}$ model performance on the GLUE benchmark validation set (V) and test set (T). Lines with † and ‡ indicate results taken from Guo et al. (2020) and Houlsby et al. (2019) (respectively).

bias terms $\mathbf{b}^{(\cdot)}$, we achieve transfer learning performance which is comparable (and sometimes better!) than fine-tuning of the entire network,

We also show that we can fine-tune only a subset of the bias parameters, namely those associated with the *query* and the *second MLP layer* (only $\mathbf{b}_q^{(\cdot)}$ and $\mathbf{b}_{m2}^{(\cdot)}$), and still achieve accuracies that rival full-model fine-tuning.

4 Experiments and Results

Datasets. We evaluate BitFit on the GLUE benchmark (Wang et al., 2018).[3] Consistent with previous work (Houlsby et al., 2019; Guo et al., 2020) we exclude the WNLI task, on which BERT models do not outperform the majority baseline.

Models and Optimization. We use the publicly available pre-trained BERT$_{\text{BASE}}$, BERT$_{\text{LARGE}}$ (Devlin et al., 2018) and RoBERTa$_{\text{BASE}}$ (Liu et al., 2019) models, using the HuggingFace (Wolf et al., 2020) interface and implementation.

Appendix §A.2 lists optimization details.

Comparison to Diff-Pruning and Adapters (Table 1) In the first experiment, we compare BitFit to Diff-Pruning method and Adapters method, when using a fewer number of parameters. Table 1 reports the dev-set and test-set performance compared to the Diff-Pruning and Adapters numbers reported by Guo et al. (2020) and Houlsby et al. (2019) (respectively). This experiment used the BERT$_{\text{LARGE}}$ model.

On validation set, BitFit outperforms Diff-Pruning on 4 out of 9 tasks, while using 6x fewer trainable parameters [4]. As for test-set results, two clear wins compared to Diff-Pruning and 4 clear wins compared to Adapters while using 45x fewer trainable parameters.

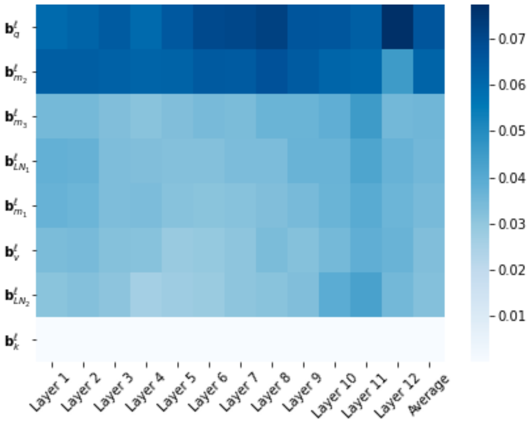


Figure 1: Change in bias components (RTE task).

Different Base-models (Table 2) We repeat the BERT$_{\text{LARGE}}$ results on different base-models (the smaller BERT$_{\text{BASE}}$ and the better performing RoBERTa$_{\text{BASE}}$). The results in Table 2 show that the trends remain consistent.

Are bias parameters special? Are the bias parameters special, or will any random subset do? We randomly sampled the same amount of parameters as in BitFit from the entire model, and fine-tuned only them ("rand uniform" line in Table 3). The results are substantially worse across all tasks; similar patterns are observed when the random parameters are sampled as complete rows/columns in the parameter matrices ("rand row/col" line in Table 3).

Fewer bias parameters (Table 3) Can we fine-tune on only a subset of the bias-parameter?

We define the amount of change in a bias vector $\mathbf{b}$ to be $\frac{1}{\mathbf{dim}(\mathbf{b})}\|\mathbf{b}_0 - \mathbf{b}_F\|_1$, that is, the average absolute change, across its dimensions, between the initial LM values $\mathbf{b}_0$ and its fine-tuned values $\mathbf{b}_F$. Figure 1 shows the change per bias term and layer, for the RTE task (other tasks look very similar, see Appendix §A.4). The 'key' bias $\mathbf{b}_k$ has zero

[3] Appendix §A.3 lists the tasks and evaluation metrics.

[4] QNLI results are not directly comparable, as the GLUE benchmark updated the test set since then.

	Method	%Param	QNLI	SST-2	MNLI$_{\text{m}}$	MNLI$_{\text{mm}}$	CoLA	MRPC	STS-B	RTE	QQP	Avg.
BB	Full-FT	100%	**90.7±0.2**	92.0±0.4	**83.5±0.1**	**83.7±0.3**	56.4±0.9	89.0±1.0	88.9±0.7	70.5±0.6	**87.1±0.1**	82.3
BB	BitFit	0.09%	90.2±0.2	**92.1±0.3**	81.4±0.2	82.2±0.2	**58.8±0.5**	**90.4±0.5**	**89.2±0.2**	**72.3±0.9**	84.0±0.2	**82.4**
BL	Full-FT	100%	**91.7±0.1**	**93.4±0.2**	**85.5±0.4**	**85.7±0.4**	62.2±1.2	90.7±0.3	90.0±0.4	71.9±1.3	**87.5±0.4**	84.1
BL	BitFit	0.08%	91.4±2.4	93.2±0.4	84.4±0.2	84.8±0.1	**63.6±0.7**	**91.7±0.5**	**90.3±0.1**	**73.2±3.7**	85.4±0.1	**84.2**
Ro	Full-FT	100%	**92.3±0.2**	**94.2±0.4**	**86.4±0.3**	**86.9±0.3**	61.1±0.8	**92.5±0.4**	90.6±0.2	77.4±1.0	**88.0±0.2**	**85.3**
Ro	BitFit	0.09%	91.3±0.2	93.7±0.1	84.8±0.1	85.2±0.2	**61.8±1.3**	92.0±0.4	**90.8±0.3**	**77.8±1.7**	84.5±0.2	84.6

Table 2: Dev-set results for different base models. **BB**: BERT$_{\text{BASE}}$. **BL**: BERT$_{\text{LARGE}}$. **Ro**: RoBERTa$_{\text{BASE}}$.

	% Param	QNLI	SST-2	MNLI$_{\text{m}}$	MNLI$_{\text{mm}}$	CoLA	MRPC	STS-B	RTE	QQP	Avg.
Full-FT	100%	**90.7±0.2**	92.0±0.4	**83.5±0.1**	**83.7±0.3**	56.4±0.9	89.0±1.0	88.9±0.7	70.5±0.6	**87.1±0.1**	82.3
BitFit	0.09%	90.2±0.2	**92.1±0.3**	81.4±0.2	82.2±0.2	**58.8±0.5**	**90.4±0.5**	**89.2±0.2**	**72.3±0.9**	84.0±0.2	**82.4**
$\mathbf{b}_{m2}, \mathbf{b}_q$	0.04%	89.4±0.1	91.2±0.2	80.4±0.2	81.5±0.2	57.4±0.8	89.0±0.2	88.4±0.1	68.6±0.6	83.7±0.2	81.1
$\mathbf{b}_{m2}$	0.03%	88.9±0.1	91.1±0.3	79.9±0.3	80.7±0.2	54.9±0.9	87.9±0.6	88.2±0.1	66.8±0.6	82.1±0.4	80.0
$\mathbf{b}_q$	0.01%	86.8±0.1	89.6±0.2	74.4±0.3	75.7±0.2	49.1±1.5	84.4±0.2	85.6±0.1	61.4±1.1	80.6±0.4	76.6
Frozen	0.0%	68.7±0.3	81.7±0.1	42.4±0.1	43.8±0.1	31.9±1.1	81.1±0.1	71.4±0.1	56.9±0.4	62.4±0.2	62.1
rand uniform	0.09%	87.8±0.3	90.5±0.3	78.3±0.3	78.8±0.2	54.1±1.0	84.3±0.3	87.2±0.4	62.9±0.9	82.4±0.3	78.5
rand row/col	0.09%	88.4±0.2	91.0±0.3	79.4±0.3	80.1±0.3	53.4±0.6	88.0±0.7	87.9±0.2	65.1±0.7	82.3±0.2	79.5

Table 3: Fine-tuning using a subset of the bias parameters. Reported results are for the BERT$_{\text{BASE}}$ model.

change, consistent with the theoretical observation in Cordonnier et al. (2020). In contrast, $\mathbf{b}_q$, the bias of the queries, and $\mathbf{b}_{m2}$, the bias of the intermediate MLP layers (which take the input from 768-dims to 3072), change the most. Table 3 reports dev-set results when fine-tuning only the $\mathbf{b}_q^{(\cdot)}$ and $\mathbf{b}_{m2}^{(\cdot)}$ bias terms, for the BERT$_{\text{BASE}}$ model. Results are only marginally lower than when tuning all bias parameters. Tuning either $\mathbf{b}_q^{(\cdot)}$ or $\mathbf{b}_{m2}^{(\cdot)}$ alone yields substantially worse results, indicating both bias types are essential. As expected, using a frozen BERT$_{\text{BASE}}$ model yields much worse results.

Generalization gap. While in most cases full fine-tuning reaches nearly 100% train accuracy, we find that the generalization gap (Shalev-Shwartz and Ben-David, 2014)—the difference between training error and test error—is substantially smaller for the BitFit models.

Token-level tasks. The GLUE tasks are all sentence level. We also experimented with token-level PTB POS-tagging. Full-FT results for BERT$_{\text{BASE}}$, BERT$_{\text{LARGE}}$ and RoBERTa$_{\text{BASE}}$ are 97.2, 97.4, 97.2, while BitFit results are 97.2, 97.4, 97.1.

Size of training data. The GLUE results suggest a reverse correlation between BitFit ability to reach Full-FT performance, and training set size. To test this (and to validate another token-level task), we train on increasing-sized subsets of SQuAD v1.0 Rajpurkar et al. (2016a). The results on Figure 2 show a clear trend: BitFit dominates over Full-FT in the smaller-data regime, while the trend is reversed when more training data is available. We conclude that BitFit is a worthwhile targetted fine-tuning method in small-to-medium data regimes.

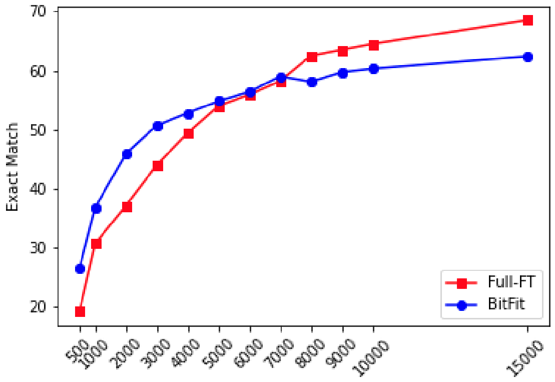


Figure 2: Comparison of BitFit and Full-FT with BERT$_{\text{BASE}}$ exact match score on SQuAD validation set.

5 Related Work

The problem of identifying the minimal set of parameters that need to be fine-tuned to achieve good performance in end-tasks relates both to practical questions of model compression, and also to more fundamental question on the nature of the pre-training and finetuning process, the “linguistic knowledge“ induced by each of them, and the extent to which it generalizes to different tasks.

Over-parameterization Large LM models were shown to be *over-parameterized*: they contain more parameters than needed in inference (Buciluă et al., 2006; Hinton et al., 2015; Urban et al., 2017; Karnin, 1990; Reed, 1993; Augasta and Kathirvalavakumar, 2013; Liu et al., 2014; Han et al., 2015; Molchanov et al., 2017). Gordon et al. (2020) have demonstrated that overparmeterization can be exploited in finetuning: pruned network perform well in transfer setting. We work in a complementary setting, where the entire model is kept, but only some parameters are updated. The remarkable success of those works have sparked interest the lottery-ticket hypothesis (Frankle and Carbin, 2019; Chen et al., 2020; Prasanna et al., 2020): the conjecture that large models are needed in pretraining only to induce (in high probability) the existing of sub-networks initialized with the correct inductive bias for learning, and the findings that those sparse networks often transfer well to different tasks.

Bias terms Bias terms and their importance are rarely discussed in the literature.[5] Zhao et al. (2020) describe a masking-based fine-tuning method, and explicitly mention *ignoring* the bias terms, as handling them "did not observe a positive effect on performance".

An exception is the work of Wang et al. (2019) who analyzed bias terms from the perspective of attribution method. They demonstrate that the last layer bias values are responsible for the predicted class, and propose a way to back-propagate their importance. Michel and Neubig (2018) fine-tuned the biases of the output softmax in an NMT systems, to personalize the output vocabulary, and Frankle et al. (2020) have demonstrated that randomly-initialized CNNs achieve reasonable accuracy after training the batch-norm layers alone. Finally, and closest to our work, Cai et al. (2020) demonstrate that bias-only fine-tuning similar to ours is effective also for adaptation of pre-trained computer vision models. Our work empirically shows the importance and power of the bias parameters to substantially change the networks' behavior, calling for further analysis and attention on the bias terms.

6 Conclusions

We propose BitFit, a novel method for localized, fast fine-tuning of pre-trained transformers for end-tasks. The method focuses the finetuning on a specific fraction of the model parameters—the biases—and maintains good performance in all GLUE tasks we evaluated on. The focus on modifying a small group of parameters eases deployment, as the vast majority of the parameters of the model are shared between various NLP tasks. It also allows for efficient hardware implementations that hard-wire most of the network computation with the pre-trained weights, while only allowing few changeable parts for inference time.

Besides its empirical utility, the remarkable effectiveness of bias-only fine-tuning raises intriguing questions on the fine-tuning dynamics of pre-trained transformers, and the relation between the bias terms and transfer between LM and new tasks.

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement No. 802774 (iEXTRACT).

References

M. Gethsiyal Augasta and T. Kathirvalavakumar. 2013. Pruning algorithms of neural networks - a comparative study. *Central Eur. J. Comput. Sci.*, 3(3):105–115.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Han Cai, Chuang Gan, Ligeng Zhu, and Song Han. 2020. Tiny transfer learning: Towards memory-efficient on-device learning. *CoRR*, abs/2007.11622.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained BERT networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. 2020. Multi-head attention: Collaborate instead of concatenate. *CoRR*, abs/2006.16362.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

[5] Indeed, the equations in the paper introducing the Transformer model (Vaswani et al., 2017) do not include bias terms at all, and their existence in the BERT models might as well be a fortunate mistake.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Jonathan Frankle, David J. Schwab, and Ari S. Morcos. 2020. Training batchnorm and only batchnorm: On the expressive power of random features in cnns. *CoRR*, abs/2003.00152.

Mitchell A. Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: studying the effects of weight pruning on transfer learning. *CoRR*, abs/2002.08307.

Demi Guo, Alexander M. Rush, and Yoon Kim. 2020. Parameter-efficient transfer learning with diff pruning.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28:1135–1143.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. *CoRR*, abs/1902.00751.

Shankar Iyer, Nikhil Dandekar, and Kornel Csernai. 2017. First quora dataset release: Question pairs.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2019. Spanbert: Improving pre-training by representing and predicting spans. *CoRR*, abs/1907.10529.

Ehud D. Karnin. 1990. A simple procedure for pruning back-propagation trained neural networks. *IEEE Trans. Neural Networks*, 1(2):239–242.

Chao Liu, Zhiyong Zhang, and Dong Wang. 2014. Pruning deep neural networks by optimal brain damage. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 1092–1095. ISCA.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Paul Michel and Graham Neubig. 2018. Extreme adaptation for personalized neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 312–318. Association for Computational Linguistics.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2017. Pruning convolutional neural networks for resource efficient inference. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines.

Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When BERT plays the lottery, all tickets are winning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3208–3229. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016a. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016b. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Russell Reed. 1993. Pruning algorithms-a survey. *IEEE Trans. Neural Networks*, 4(5):740–747.

Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Gregor Urban, Krzysztof J. Geras, Samira Ebrahimi Kahou, Özlem Aslan, Shengjie Wang, Abdelrahman Mohamed, Matthai Philipose, Matthew Richardson, and Rich Caruana. 2017. Do deep convolutional nets really need to be deep and convolutional? In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461.

Shengjie Wang, Tianyi Zhou, and Jeff A. Bilmes. 2019. Bias also matters: Bias attribution for deep neural network explanation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6659–6667. PMLR.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. 2020. Masking as an efficient alternative to finetuning for pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2226–2241, Online. Association for Computational Linguistics.

A Appendices

A.1 Layer naming

For convenience, we relate the notation used in the paper with the names of the corresponding parameters in the popular HuggingFace (Wolf et al., 2020) implementation.

HuggingFace Parameter Name	BitFit notation
attention.self.query.bias	$\mathbf{b}_q$
attention.self.key.bias	$\mathbf{b}_k$
attention.self.value.bias	$\mathbf{b}_v$
attention.output.dense.bias	$\mathbf{b}_{m_1}$
attention.output.LayerNorm.bias	$\mathbf{b}_{LN_1}$
intermediate.dense.bias	$\mathbf{b}_{m_2}$
output.dense.bias	$\mathbf{b}_{m_3}$
output.LayerNorm.bias	$\mathbf{b}_{LN_2}$

Table 4: Mapping the HuggingFace's BertLayer bias parameters names to BitFit paper bias notation.

A.2 Training Details

To perform classification with BERT, we follow the approach of Devlin et al. (2018), and attach a linear layer to the contextual embedding of the [CLS] token to predict the label. The GLUE tasks are fed into BERT using the standard procedures.

We optimize using AdamW (Loshchilov and Hutter, 2017), with batch sizes of 16. For full fine-tuning, we used initial learning rates in {1e-5, 2e-5, 3e-5, 5e-5}, and for the bias-only experiments we used initial learning rates in {1e-4, 4e-4, 7e-4, 1e-3} as the smaller rates took a very long time to converge on some of the tasks. With the larger learning rates, the bias-only fine-tuning converged in 8 or fewer epochs for most tasks, and up to 20 epochs on the others. We did not perform hyper-parameter optimization beyond the minimal search over 4 learning rates. In each evaluation we report X±Y where X is the average result for training 5 models with 5 different random seeds, Y is the standard deviation.

To perform classification with RoBERTa$_{\text{BASE}}$, we follow the above details but without hyperparameter search over the learning rates, for bias-only fine-tuning we used 1e-4 as learning rate and for full fine-tuning we used 1e-5 as learning rate.

As Mosbach et al. (2020) show, fine-tuning BERT$_{\text{LARGE}}$ and RoBERTa$_{\text{BASE}}$ is a unstable due to vanishing gradients. BitFit allows for the usage of bigger learning rates, and overall the optimization process is much more stable, when compared with a full fine-tuning.

Task Name	Metric
QNLI	acc.
SST-2	acc.
MNLI	matched acc./mismatched acc.
CoLA	Matthews corr.
MRPC	F1
STS-B	Spearman corr.
RTE	acc.
QQP	F1

Table 5: Metrics that we use to evaluate GLUE Benchmark.

Task Name	BERT$_{\text{BASE}}$	BERT$_{\text{LARGE}}$
QNLI	1e-4	7e-4
SST-2	4e-4	4e-4
MNLI	1e-4	1e-4
CoLA	7e-4	4e-4
MRPC	7e-4	1e-3
STS-B	1e-4	1e-4
RTE	1e-3	4e-4
QQP	4e-4	4e-4

Table 6: Learning rate configurations for best performing models.

A.3 GLUE Benchmark

We provide information on the GLUE tasks we evaluated on, as well as on the evaluation metrics. We test our approach on the following subset of the GLUE (Wang et al., 2018) tasks: The Corpus of Linguistic Acceptability (CoLA; Warstadt et al. (2018)), The Stanford Sentiment Treebank (SST-2; Socher et al. (2013)), The Microsoft Research Paraphrase Corpus (MRPC; Dolan and Brockett (2005)), The Quora Question Pairs (QQP; Iyer et al. (2017)), The Semantic Textual Similarity Benchmark (STS-B; Cer et al. (2017)), The Multi-Genre Natural Language Inference Corpus (MNLI; Bowman et al. (2015)), The Stanford Question Answering Dataset (QNLI; Rajpurkar et al. (2016b)) and The Recognizing Textual Entailment (RTE; Dagan et al. (2005)).

The metrics that we used to evaluate GLUE Benchmark are in Table 5. Learning rate configurations for best performing models are in Table 6. For all the experiments we used the common train:dev:test partition of GLUE.

A.4 Amount of change in bias terms

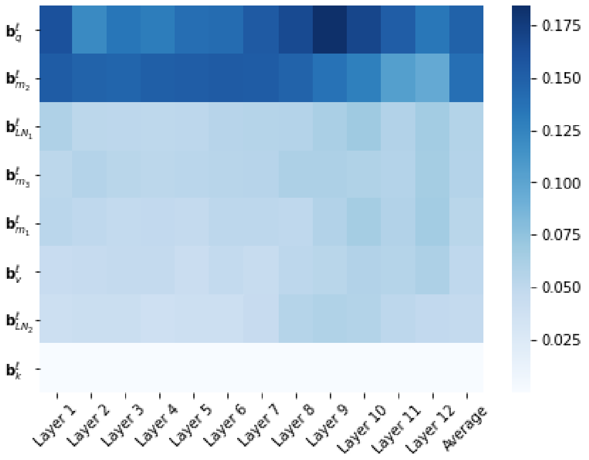


Figure 3: Change in bias components (CoLA task).

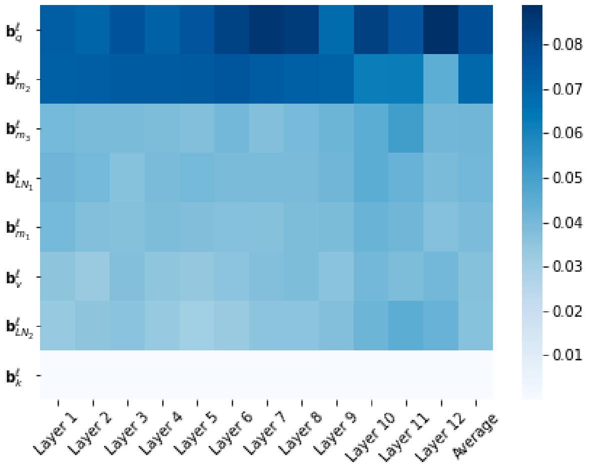


Figure 4: Change in bias components (MRPC task).

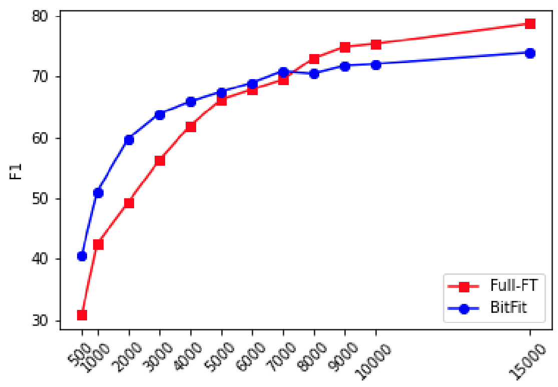


Figure 6: Comparison of BitFit and Full-FT with $\text{BERT}_{\text{BASE}}$ F1 score on SQuAD validation set.

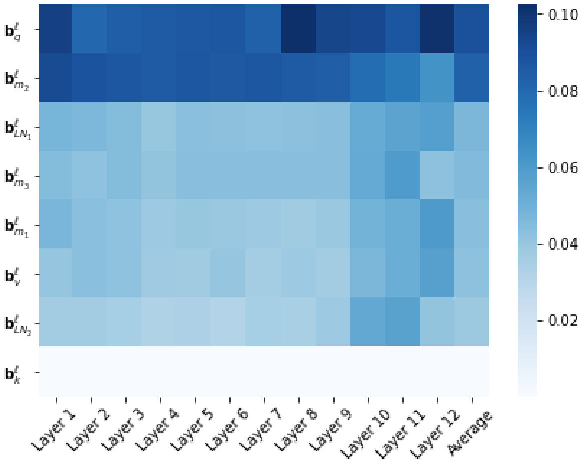


Figure 5: Change in bias components (STS-B task).

A.5 SQuAD F1 Results

Are Shortest Rationales the Best Explanations for Human Understanding?

Hua Shen[†] **Tongshuang Wu**[◇] **Wenbo Guo**[†] **Ting-Hao 'Kenneth' Huang**[†]
[†]College of Information Sciences and Technology, Pennsylvania State University
[◇]Paul G. Allen School of Computer Science and Engineering, University of Washington
`{huashen218,wzg13,txh710}@psu.edu`
`wtshuang@cs.washington.edu`

Abstract

Existing self-explaining models typically favor extracting the shortest possible rationales — snippets of an input text "responsible for" corresponding output — to explain the model prediction, with the assumption that shorter rationales are more intuitive to humans. However, this assumption has yet to be validated. Is the shortest rationale indeed the most human-understandable? To answer this question, we design a self-explaining model, LimitedInk, which allows users to extract rationales at any target length. Compared to existing baselines, LimitedInk achieves compatible end-task performance and human-annotated rationale agreement, making it a suitable representation of the recent class of self-explaining models. We use LimitedInk to conduct a user study on the impact of rationale length, where we ask human judges to predict the sentiment label of documents based only on LimitedInk-generated rationales with different lengths. We show rationales that are too short do not help humans predict labels better than randomly masked text, suggesting the need for more careful design of the best human rationales.[1]

1 Introduction

While neural networks have recently led to large improvements in NLP, most of the models make predictions in a black-box manner, making them indecipherable and untrustworthy to human users. In an attempt to faithfully explain model decisions to humans, various work has looked into extracting *rationales* from text inputs (Jain et al., 2020; Paranjape et al., 2020), with *rationale* defined as the "shortest yet sufficient subset of input to predict the same label" (Lei et al., 2016; Bastings et al., 2019). The underlying assumption is two-fold: (1) by retaining the label, we are extracting the texts used by predictors (Jain et al., 2020); and (2) short rationales are more readable and intuitive for end-users, and thus preferred for human understanding (Vafa et al., 2021). Importantly, prior work has knowingly traded off some amount of model performance to achieve the shortest possible rationales. For example, when using less than 50% of text as rationales for predictions, Paranjape et al. (2020) achieved an accuracy of 84.0% (compared to 91.0% if using the full text). However, the assumption that the shortest rationales have better human interpretability has not been validated by

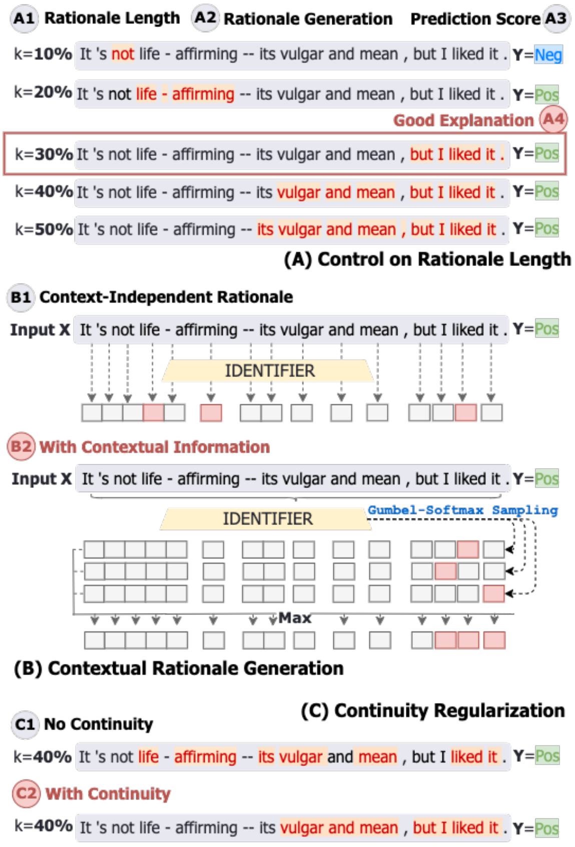

A1 Rationale Length
A2 Rationale Generation
Prediction Score A3
k=10% It 's not life - affirming -- its vulgar and mean , but I liked it . Y=Neg
k=20% It 's not life - affirming -- its vulgar and mean , but I liked it . Y=Pos
Good Explanation A4
k=30% It 's not life - affirming -- its vulgar and mean , but I liked it . Y=Pos
k=40% It 's not life - affirming -- its vulgar and mean , but I liked it . Y=Pos
k=50% It 's not life - affirming -- its vulgar and mean , but I liked it . Y=Pos
(A) Control on Rationale Length
B1 Context-Independent Rationale
Input X It 's not life - affirming -- its vulgar and mean , but I liked it . Y=Pos
IDENTIFIER
B2 With Contextual Information
Input X It 's not life - affirming -- its vulgar and mean , but I liked it . Y=Pos
IDENTIFIER
Gumbel-Softmax Sampling
Max
(B) Contextual Rationale Generation
(C) Continuity Regularization
C1 No Continuity
k=40% It 's not life - affirming -- its vulgar and mean , but I liked it . Y=Pos
C2 With Continuity
k=40% It 's not life - affirming -- its vulgar and mean , but I liked it . Y=Pos

Figure 1: LimitedInk's rationale generation with length control: (A) control rationale generation with different lengths; (B) incorporating contextual information into rationale generation; (C) regularizing continuous rationale for human interpretability. Examples use the SST dataset for sentiment analysis (Socher et al., 2013).

[1] Find open-source code at: `https://github.com/huashen218/LimitedInk.git`

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 10 - 19
May 22-27, 2022 ©2022 Association for Computational Linguistics

human studies (Shen and Huang, 2021). Moreover, when the rationale is too short, the model has much higher chance of missing the main point in the full text. In Figure 1*A*, although the model can make the correct positive prediction when using only 20% of the text, it relies on a particular adjective, “life-affirming,” which is seemingly positive but does not reflect the author’s sentiment. These rationales may be confusing when presented to end-users.

In this work, we ask: *Are shortest rationales really the best for human understanding?* To answer the question, we first design LimitedInk, a self-explaining model that flexibly extracts rationales at any target length (Figure 1*A*). LimitedInk allows us to control and compare rationales of varying lengths on input documents. Besides **controls on rationale length**, we also design LimitedInk’s sampling process and objective function to be **context-aware** (*i.e.,* rank words based on surrounding context rather than individually, Figure 1B_2) and **coherent** (*i.e.,* prioritize continuous phrases over discrete tokens, Figure 1C_2). Compared to existing baselines (*e.g.,* `Sparse-IB`), LimitedInk achieves compatible end-task performance and alignment with human annotations on the ERASER (DeYoung et al., 2020) benchmark, which means it can represent recent class of self-explaining models.

We use LimitedInk to conduct user studies to investigate the effect of rationale length on human understanding. Specifically, we ask MTurk participants to predict document sentiment polarities based on only LimitedInk-extracted rationales. By contrasting rationales at five different length levels, we find that shortest rationales are largely not the best for human understanding. In fact, humans do not perform better prediction accuracy and confidence better than using randomly masked texts when rationales are too short (*e.g.,* 10% of input texts). In summary, this work encourages a rethinking of self-explaining methods to find the right balance between brevity and sufficiency.

2 LimitedInk

2.1 Self-Explaining Model Definition

We start by describing typical self-explaining methods (Lei et al., 2016; Bastings et al., 2019; Paranjape et al., 2020). Consider a text classification dataset containing each document input as a tuple $(\mathbf{x}, y)$. Each input $\mathbf{x}$ includes n features (*e.g.,* sentences or tokens) as $\mathbf{x} = [x_1, x_2, ..., x_n]$, and y is the prediction. The model typically consists of an *identifier* $\mathbf{idn}(\cdot)$ to derive a boolean mask $\mathbf{m} = [m_1, m_2, ..., m_n]$, where $m_i \in \{1, 0\}$ indicates whether feature x_i is in the rationale or not. Note that the mask $\mathbf{m}$ is typically a binary selection from the *identifier*’s probability distribution, i.e., $\mathbf{m} \sim \mathbf{idn}(\mathbf{x})$. Then it extracts rationales $\mathbf{z}$ by $\mathbf{z} = \mathbf{m} \odot \mathbf{x}$, and further leverages a *classifier* $\mathbf{cls}(\cdot)$ to make a prediction y based on the identified rationales as $y = \mathbf{cls}(\mathbf{z})$. The optimization objective is:

$$\min_{\theta_{\mathbf{idn}},\theta_{\mathbf{cls}}} \underbrace{\mathbb{E}_{\mathbf{z}\sim\mathbf{idn}(\mathbf{x})}\mathcal{L}(\mathbf{cls}(\mathbf{z}), y)}_{\text{sufficient prediction}} + \underbrace{\lambda\Omega(\mathbf{m})}_{\text{regularization}} \qquad (1)$$

where $\theta_{\mathbf{idn}}$ and $\theta_{\mathbf{cls}}$ are trainable parameters of *identifier* and *classifier*. $\Omega(\mathbf{m})$ is the regularization function on mask and λ is the hyperparameter.

2.2 Generating Length Controllable Rationales with Contextual Information

We next elaborate on the definition and method of controlling rationale length in LimitedInk Assuming that the rationale length is k as prior knowledge, we enforce the generated boolean mask to sum up to k as $k = \sum_{i=1}^{n}(m_i)$, where $\mathbf{m} = \mathbf{idn}(\mathbf{x}, k)$. Existing self-explaining methods commonly solve this by sampling from a Bernoulli distribution over input features, thus generating each mask element m_i independently conditioned on each input feature x_i (Paranjape et al., 2020). For example, in Figure 1B_1), “life affirming” is selected independent of the negation context “not” before it, which contradicts with the author’s intention. However, these methods potentially neglect the contextual input information. We leverage the concrete relaxation of subset sampling technique (Chen et al., 2018) to incorporate contextual information into rationale generation process (see Figure 1B_2), where we aim to select the top-k important features over all n features in input $\mathbf{x}$ via Gumbel-Softmax Sampling (*i.e.,* applying the Gumbel-softmax trick to approximate weighted subset sampling process). To further guarantee precise rationale length control, we deploy the *vector and sort* regularization on mask $\mathbf{m}$ (Fong et al., 2019). See more model details in Appendix A.1.

2.3 Regularizing Rationale Continuity

To further enforce coherent rationale for human interpretability, we employ the Fused Lasso to encourage continuity property (Jain et al., 2020; Bastings et al., 2019). The final mask regularization is:

$$\Omega(\mathbf{m}) = \lambda_1 \underbrace{\sum_{i=1}^{n} |m_i - m_{i-1}|}_{\text{Continuity}} + \lambda_2 \underbrace{\| \text{vecsort}\,(m) - \hat{m} \|}_{\text{Length Control}} \qquad (2)$$

Method	Movies				BoolQ				Evidence Inference				MultiRC				FEVER			
	Task	P	R	F1	Task	P	R	F1	Task	P	R	F1	Task	P	R	F1	Task	P	R	F1
`Full-Text`	.91	-	-	-	.47	-	-	-	.48	-	-	-	.67	-	-	-	.89	-	-	-
`Sparse-N`	.79	.18	.36	.24	.43	.12	.10	.11	.39	.02	.14	.03	.60	.14	.35	.20	.83	.35	.49	.41
`Sparse-C`	.82	.17	.36	.23	.44	.15	.11	.13	.41	.03	.15	.05	.62	.15	**.41**	.22	.83	.35	.52	.42
`Sparse-IB`	.84	.21	.42	.28	.46	**.17**	.15	.15	.43	.04	.21	.07	.62	.20	.33	.25	.85	**.37**	.50	**.43**
LimitedInk	**.90**	**.26**	**.50**	**.34**	**.56**	.13	**.17**	**.15**	**.50**	**.04**	**.27**	**.07**	**.67**	**.22**	.40	**.28**	**.90**	.28	**.67**	.39
Length Level	50%				30%				50%				50%				40%			

Table 1: LimitedInk performs compatible with baselines in terms of end-task performance (**Task**, weighted average F1) and human annotated rationale agreement (**P**recision, **R**ecall, **F1**). All results are on test sets and are averaged across five random seeds. For LimitedInk, we report results for the best performing *length level*.

For BERT-based models, which use subword-based tokenization algorithms (*e.g.,* WordPiece), we assign each token's importance score as its subtokens' maximum score to extract rationales during model inference (see Figure 1*C*).

3 Model Performance Evaluation

We first validate LimitedInk on two common rationale evaluation metrics, including end-task performance and human annotation agreement.

3.1 Experimental Setup

We evaluate our model on five text classification datasets from the ERASER benchmark (DeYoung et al., 2020). We design the *identifier* module in LimitedInk as a BERT-based model, followed by two linear layers with the ReLU function and dropout technique. The temperature for Gumbel-softmax approximation is fixed at 0.1. Also, we define the *classifier* module as a BERT-based sequence classification model to predict labels. We train five individual self-explaining models of different rationale lengths with training and validation sets, where we set the rationale lengths as {10%, 20%, 30%, 40%, 50%} of all input text. Then we select one out of the five models, which has the best weighted average F1 score, to compare with current baselines on end-task performance and human annotation agreement on test sets. Note that we use all models with five rationale lengths in human evaluation described in Section 4.

Baselines. We compare LimitedInk with four baselines. `Full-Text` consists of only the *classifier* module with full-text inputs. `Sparse-N` enforces shortest rationales by minimizing rationale mask length (Lei et al., 2016; Bastings et al., 2019). `Sparse-C` controls rationale length by penalizing the mask when its length is less than a threshold (Jain et al., 2020). `Sparse-IB` enables length control by minimizing the KL-divergence between the generated mask with a prior distribution (Paranjape et al., 2020). See Appendix A.1 for more model and baseline details.

3.2 Evaluation Results

End-Task Performance. Following metrics in DeYoung et al. (2020), we report the weighted average F1 scores for end-task classification performance. Among five LimitedInk models with different rationale lengths, Table 1 reports the model with the best end-task performance on the test set. We observe that LimitedInk performs similarly to or better than the self-explaining baselines in all five datasets. See ablation studies in Appendix A.2.

Human-Annotated Rationale Agreement. We calculate the alignment between generated rationales and human annotations collected in the ERASER benchmark (DeYoung et al., 2020). As also shown in Table 1, we report the Token-level F1 (F1) metric along with corresponding Precision (P) and Recall (R) scores. The results show that LimitedInk can generate rationales that are consistent with human annotations and comparable to self-explaining baselines in all datasets.

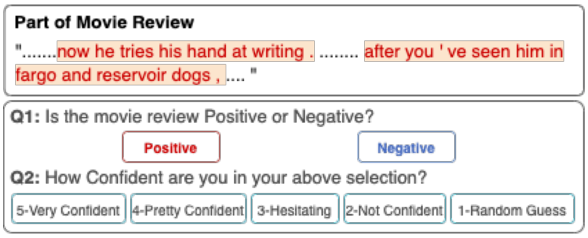
4 Human Evaluation

Equipped with LimitedInk, we next carry out human studies to investigate the effect of rationale length on human understanding.

4.1 Study Design

Our goal is to quantify human performance on predicting the labels and confidence based solely on the rationales with different lengths. To do so, we control LimitedInk to extract rationales of different lengths, and recruit Mechanical Turk (MTurk) workers to provide predictions and confidence.

Dataset & rationale extraction. We focus on sentiment analysis in user study, and randomly sample 100 reviews from the Movie Reviews (Zaidan



Part of Movie Review

".......now he tries his hand at writing . ... after you ' ve seen him in fargo and reservoir dogs , ... "

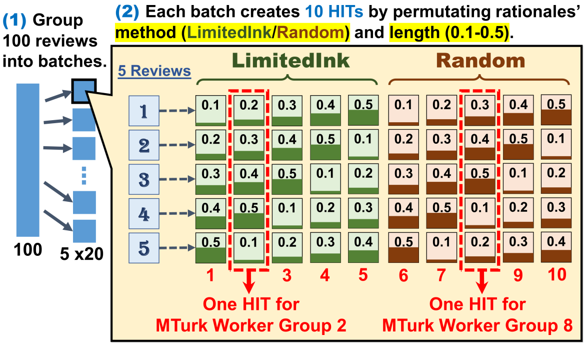
Q1: Is the movie review Positive or Negative?

Positive | Negative

Q2: How Confident are you in your above selection?

5-Very Confident | 4-Pretty Confident | 3-Hesitating | 2-Not Confident | 1-Random Guess

Figure 2: Key components of the User Interface in the MTurk *task* HITs. Note that each HIT contains five reviews with different rationale lengths.


(1) Group 100 reviews into batches.
(2) Each batch creates 10 HITs by permutating rationales' method (LimitedInk/Random) and length (0.1-0.5).
100
5 x20
5 Reviews
LimitedInk
Random
One HIT for MTurk Worker Group 2
One HIT for MTurk Worker Group 8

	1	2	3	4	5	6	7	8	9	10
1	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
2	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5	0.1
3	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5	0.1	0.2
4	0.4	0.5	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3
5	0.5	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4

Figure 3: The human evaluation's workflow. We (1) divide 100 movie reviews into 20 batches and (2) produce 10 HITs from each batch for ten worker groups.

and Eisner, 2008) test set that have correct model predictions. Then, we extract five rationales for each review using **LimitedInk**, with lengths from 10% to 50%, with an increment of 10%.

Since human accuracy likely increases when participants see more words (*i.e.,* when the lengths of rationales increase), we also create a **Random** rationale baseline, where we randomly select words of the same rationale length on the same documents (10% to 50%) while taking the continuity constraint into consideration. More details of Random baseline generation are in Appendix A.3.1.

Study Procedure. The study is completed in two steps. First, we posted a *qualification* Human Intelligence Tasks (HITs, $0.50 per assignment) on MTurk to recruit 200 qualified workers.[2] Next, the 200 recruited workers can participate the *task* HIT ($0.20 per assignment, 7 assignments posted) which contains five distinct movie reviews, with varying rationale lengths (10%-50%). In *task* HIT, as key components shown in Figure 2, we only display the rationales and mask all other words with ellipses of random length, such that participants can not infer the actual review length. Then participants are asked to guess the sentiment of the full review, and provide their confidence level based on a five-point Likert Scale (Likert, 1932). The full user interface is in Appendix A.3.2.

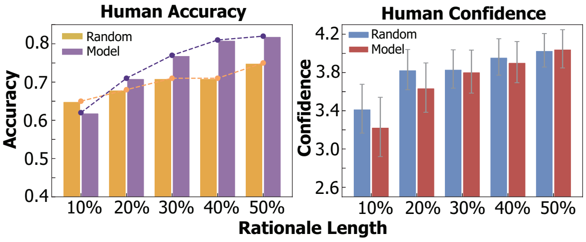


Figure 4: Human accuracy and confidence on predicting model labels given rationales with different lengths.

Participants recruiting and grouping. With each review having ten distinct rationales (five from LimitedInk and five Random), if these rationale conditions were randomly assigned, participants are likely to see the same review repeatedly and gradually see all the words. We carefully design our study to eliminate such undesired learning effect. More specifically, we group our 100 reviews into 20 batches, with five reviews in each batch (Step 1 in Figure 3). For each batch, we create five HITs for LimitedInk and Random, respectively, such that all the rationale lengths of five reviews are covered by these 10 HITs (Step 2 in Figure 3). Further, we make sure each participant is only assigned to one unique HIT, so that each participant can only see a review once. To do so, we randomly divide the 200 qualified workers into 10 worker groups (20 workers per group), and pair one worker group with only one HIT in each batch. This way, each HIT can only be accomplished by one worker group. As our participant control is more strict than regular data labeling tasks on MTurk, we keep the HITs open for 6 days. 110 out of 200 distinct workers participated in the main study, and they completed 1,169 of 1,400 assignments.

4.2 Results

We show the human prediction accuracy and confidence results in Figure 4. We find that the best explanations for human understanding are largely not the shortest rationales (10% length level): here, the human accuracy in predicting model labels is lower than for the random baseline (0.61 vs. 0.63), indicating that the shortest rationales are not the best for human understanding. There is a significant difference in human predicted labels (*i.e.,* "positive"=1,"negative"=2) between LimitedInk (M=1.24,SD=0.71) and Random

[2] In addition to our custom qualification used for worker grouping, three built-in worker qualifications are used in all of our HITs: HIT Approval Rate (≥98%), Number of Approved HITs (≥ 3000), and Locale (US Only) Qualification.

length level (%) & Extract. method		**Negative**	**Positive**
		P / R / F1	P / R / F1
10%	LimitedInk	0.66 / 0.56 / / 0.61	**0.70** / 0.58 / 0.64
	Random	**0.67 / 0.57 / 0.62**	0.66 / **0.70 / 0.68**
20%	LimitedInk	**0.75 / 0.61 / 0.67**	**0.71 / 0.77 / 0.74**
	Random	0.69 / 0.60 / 0.64	0.68 / 0.74 / 0.71
30%	LimitedInk	**0.74 / 0.76 / 0.75**	**0.81 / 0.78 / 0.79**
	Random	0.72 / 0.61 / 0.66	0.72 / 0.78 / 0.75
40%	LimitedInk	**0.84 / 0.76 / 0.80**	**0.78 / 0.85 / 0.81**
	Random	0.79 / 0.63 / 0.70	0.65 / 0.79 / 0.71
50%	LimitedInk	**0.78 / 0.78 / 0.78**	**0.85 / 0.84 / 0.85**
	Random	0.77 / 0.63 / 0.70	0.75 / 0.84 / 0.79

Table 2: Human performance (*i.e.,* Precision / Recall / F1 Score) on predicting model labels of each category in the Movie Reviews dataset.

(M=1.32,SD=0.54); t(1169)=2.27, p=0.02. Table 2 shows human performance for each category.

Additionally, notice that the slope of our model's accuracy consistently flattens as the rationale increases, whereas the random baseline does not display any apparent trend and is obviously lower than our model at higher length levels (*e.g.*, 40%). We hypothesize that this means our model is (1) indeed learning to reveal useful rationales (rather than just randomly displaying meaningless text), and (2) the amount of information necessary for human understanding only starts to saturate at around 40% of the full text. This creates a clear contrast with prior work, where most studies extract 10-30% of the text as the rationale on the same dataset (Jain et al., 2020; Paranjape et al., 2020). The eventually flattened slope potentially suggests a sweet spot to balance human understanding on rationales and sufficient model accuracy.

5 Discussion

By examining human prediction performance on five levels of rationale lengths, we demonstrate that the shortest rationales are largely not the best for human understanding. We are aware that this work has limitations. The findings are limited to Movie Reviews dataset, and we only evaluate human performance with rationales generated by the proposed LimitedInk. Still, our findings challenge the "shorter is better" assumption commonly adopted in existing self-explaining methods. As a result, we encourage future work to more cautiously define the best rationales for human understanding, and trade off between model accuracy and rationale length. More concretely, we consider that rationale models should find the right balance between brevity and sufficiency. One promising direction could be to clearly define the optimal human interpretability in a measurable way and then learn to adaptively select rationales with appropriate length.

6 Related Work

Self-explaining models. Self-explaining models, which condition predictions on their rationales, are considered more trustworthy than post-hoc explanation techniques (Rajagopal et al., 2021). However, existing efforts often enforce minimal rationale length, which degrade the predictive performance (Yu et al., 2019; Bastings et al., 2019; Jain et al., 2020). Paranjape et al. (2020) improves this by proposing an information bottleneck approach to enable rationale length control at the sentence level. In this paper, LimitedInk further enables length control at the token level to allow more flexibility needed for our human studies.

Human-grounded evaluation. A line of studies evaluated model-generated rationales by comparing them against human-annotated explanations (Carton et al., 2020; Paranjape et al., 2020). Some other studies collect feedback from users to evaluate the explanations, such as asking people to choose a preferred model (Ribeiro et al., 2016) or to guess model predictions only based on rationales (Lertvittayakumjorn and Toni, 2019; Shen and Huang, 2020).

7 Conclusion

To investigate if the shortest rationales are best understandable for humans, this work presents a self-explaining model, LimitedInk, that achieves comparable performance with current self-explaining baselines in terms of end-task performance and human annotation agreement. We further use LimitedInk to generate rationales for human studies to examine how rationale length can affect human understanding. Our results show that the shortest rationales are largely not the best for human understanding. This would encourage a rethinking of rationale methods to find the right balance between brevity and sufficiency.

8 Acknowledgment

We thank Chieh-Yang Huang for helpful comments on the paper, Bhargavi Paranjape for technical discussion of methods, and the crowd workers for participating in this study. We also thank the anonymous reviewers for their constructive feedback.

9 Ethical Considerations

This work shows that the shortest rationales are often not the best for human understanding. We thus advocate for studying how users interact with machine-generated rationales. However, we are aware that using rationales to interpret model prediction could pose some risks for users. Rationales omit a significant portion of the contents (in our case, 50% to 90% of the words in a movie review are omitted), which could convey information incorrectly or mislead users. Furthermore, machine-learned rationales could encode some unwanted biases (Chuang et al., 2021). We believe that such risks should be explicitly communicated with users in real-world applications.

References

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977, Florence, Italy. Association for Computational Linguistics.

Samuel Carton, Anirudh Rathore, and Chenhao Tan. 2020. Evaluating and characterizing human rationales. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9294–9307, Online. Association for Computational Linguistics.

Shiyu Chang, Yang Zhang, Mo Yu, and Tommi S. Jaakkola. 2020. Invariant rationalization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1448–1458. PMLR.

Jianbo Chen, Le Song, Martin J. Wainwright, and Michael I. Jordan. 2018. Learning to explain: An information-theoretic perspective on model interpretation. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 882–891. PMLR.

Yung-Sung Chuang, Mingye Gao, Hongyin Luo, James Glass, Hung-yi Lee, Yun-Nung Chen, and Shang-Wen Li. 2021. Mitigating biases in toxic language detection through invariant rationalization. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 114–120, Online. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Ruth Fong, Mandela Patrick, and Andrea Vedaldi. 2019. Understanding deep networks via extremal perturbations and smooth masks. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 2950–2958. IEEE.

Sarthak Jain, Sarah Wiegreffe, Yuval Pinter, and Byron C. Wallace. 2020. Learning to faithfully rationalize by construction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4459–4473, Online. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Piyawat Lertvittayakumjorn and Francesca Toni. 2019. Human-grounded evaluations of explanation methods for text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5195–5205, Hong Kong, China. Association for Computational Linguistics.

Rensis Likert. 1932. A technique for the measurement of attitudes. *Archives of psychology*.

Bhargavi Paranjape, Mandar Joshi, John Thickstun, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. An information bottleneck approach for controlling conciseness in rationale extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1938–1952, Online. Association for Computational Linguistics.

Dheeraj Rajagopal, Vidhisha Balachandran, Eduard H Hovy, and Yulia Tsvetkov. 2021. SELFEXPLAIN: A self-explaining architecture for neural text classifiers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 836–850, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144. ACM.

Hua Shen and Ting-Hao Huang. 2020. How useful are the machine-generated interpretations to general users? a human evaluation on guessing the incorrectly predicted labels. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 8, pages 168–172.

Hua Shen and Ting-Hao'Kenneth' Huang. 2021. Explaining the road not taken. *ACM CHI 2022 Workshop on Human-Centered Explainable AI.*

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Keyon Vafa, Yuntian Deng, David Blei, and Alexander Rush. 2021. Rationales for sequential predictions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10314–10332, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mo Yu, Shiyu Chang, Yang Zhang, and Tommi Jaakkola. 2019. Rethinking cooperative rationalization: Introspective extraction and complement control. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4094–4103, Hong Kong, China. Association for Computational Linguistics.

Omar Zaidan and Jason Eisner. 2008. Modeling annotators: A generative approach to learning from annotator rationales. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 31–40, Honolulu, Hawaii. Association for Computational Linguistics.

A Appendix

A.1 Model Details and Hyperparameters

A.1.1 Methodology Details

Concrete Relaxation of Subset Sampling Process. Given the output logits of *identifier*, we use Gumbel-softmax (Jang et al., 2017) to generate a concrete distribution as $\mathbf{c} = [c_1, ...c_n] \sim \text{Concrete}(\mathbf{idn}(\mathbf{x}))$, represented as a one-hot vector over n features where the top important feature is 1. We then sample this process k times in order to sample top-k important features, where we obtain k concrete distributions as $\{\mathbf{c}^1, ..., \mathbf{c}^k\}$. Next we define one n-dimensional random vector $\mathbf{m}$ to be the element-wise maximum of these k concrete distributions along n features, denoted as $\mathbf{m} = \max_j \{\mathbf{c}_i^j\}_{i=n}^{j=k}$. Discarding the overlapping features to keep the rest, we then use $\mathbf{m}$ as the k-hop vector to approximately select the top-k important features over document $\mathbf{x}$.

Vector and sort regularization. We deploy a *vector and sort* regularization on mask $\mathbf{m}$ (Fong et al., 2019), where we sort the output mask m in a increasing order and minimize the L_1 norm between m and a reference $\hat{m}$ consisting of $n-k$ zeros followed by k ones.

A.1.2 Model Training Details

Training and inference. During training, we select the Adam optimizer with the learning rate at 2e-5 with no decay. We set hyperparameters in Equation 5 and 2 as $\lambda = 1e-4$, $v_1 = 0.5$ and $v_2 = 0.3$ and trained 6 epochs for all models. Furthermore, we train LimitedInk on a set of sparsity levels as $k = \{10\%, 20\%, 30\%, 40\%, 50\%\}$ and choose models with optimal predictive performance on validation sets.

A.1.3 Details of Self-Explaining Baselines

We compare our method with state-of-the-art self-explaining baseline models.

Sparse-N (Minimization Norm). This method learns the short mask with minimal L_0 or L_1 norm (Lei et al., 2016; Bastings et al., 2019), which penalizes for the total number of selected words in the explanation.

$$\min \ \mathbb{E}_{\mathbf{z}\sim\mathbf{idn}(\mathbf{x})}\mathcal{L}(\mathbf{cls}(\mathbf{z}), y) + \lambda ||m|| \tag{3}$$

Sparse-C (Controlled Norm Minimization). This method controls the mask sparsity through a tunable predefined sparsity level α (Chang et al., 2020; Jain et al., 2020). The mask is penalized as below as long as the sparsity level α is passed.

$$\min \mathbb{E}_{\mathbf{z} \sim \mathbf{idn}(\mathbf{x})} \mathcal{L}(\mathbf{cls}(\mathbf{z}), y) + \lambda \max(0, \frac{||\mathbf{m}||}{N} - \alpha) \tag{4}$$

where N is the input length and $||m||$ denotes mask penalty with L_1 norm.

Sparse IB (Controlled Sparsity with Information Bottleneck). This method introduces a prior probability of $\mathbf{z}$, which approximates the marginal $p(\mathbf{m})$ of mask distribution; and $p(\mathbf{m}|\mathbf{x})$ is the parametric posterior distribution over $\mathbf{m}$ conditioned on input $\mathbf{x}$ (Paranjape et al., 2020). The sparsity control is achieved via the information loss term, which reduces the KL divergence between the posterior distribution $p(\mathbf{m}|\mathbf{x})$ that depends on $\mathbf{x}$ and a prior distribution $r(\mathbf{m})$ that is independent of $\mathbf{x}$.

$$\min \mathbb{E}_{\mathbf{z} \sim \mathbf{idn}(\mathbf{x})} \mathcal{L}(\mathbf{cls}(\mathbf{z}), y) + \lambda KL[p(\mathbf{m}|\mathbf{x}), r(\mathbf{m})] \tag{5}$$

A.2 Ablation Study on Model Components

We provide an ablation study on the Movie dataset to evaluate each loss term's influence on end-task prediction performance, including Precision, Recall, and F1 scores. The result is shown in Table 3.

Setups	**End-Task Prediction**		
	Precision	**Recall**	**F1**
No Sufficiency	0.25	0.50	0.34
No Continuity	0.82	0.81	0.81
No Sparsity	0.80	0.79	0.79
No Contextual	0.83	0.83	0.83
Our Model	**0.91**	**0.90**	**0.90**

Table 3: Ablation study of each module in our model on Movie Review dataset.

A.3 Additional Details of Human Study

A.3.1 Generating Random Baselines

Human accuracy likely increases when participants can see more words, *i.e.*, when the lengths of rationales increase. If a rationale and a random text span have the same number of words, the rationale should help readers predict the label better. We created a simple baseline that generated rationales by randomly selecting words to form the rationales. We could control (1) how many words to select and (2) how many disjointed rationales to produce. In the study, we set these two numbers to be identical to that of LimitedInk at each length level.

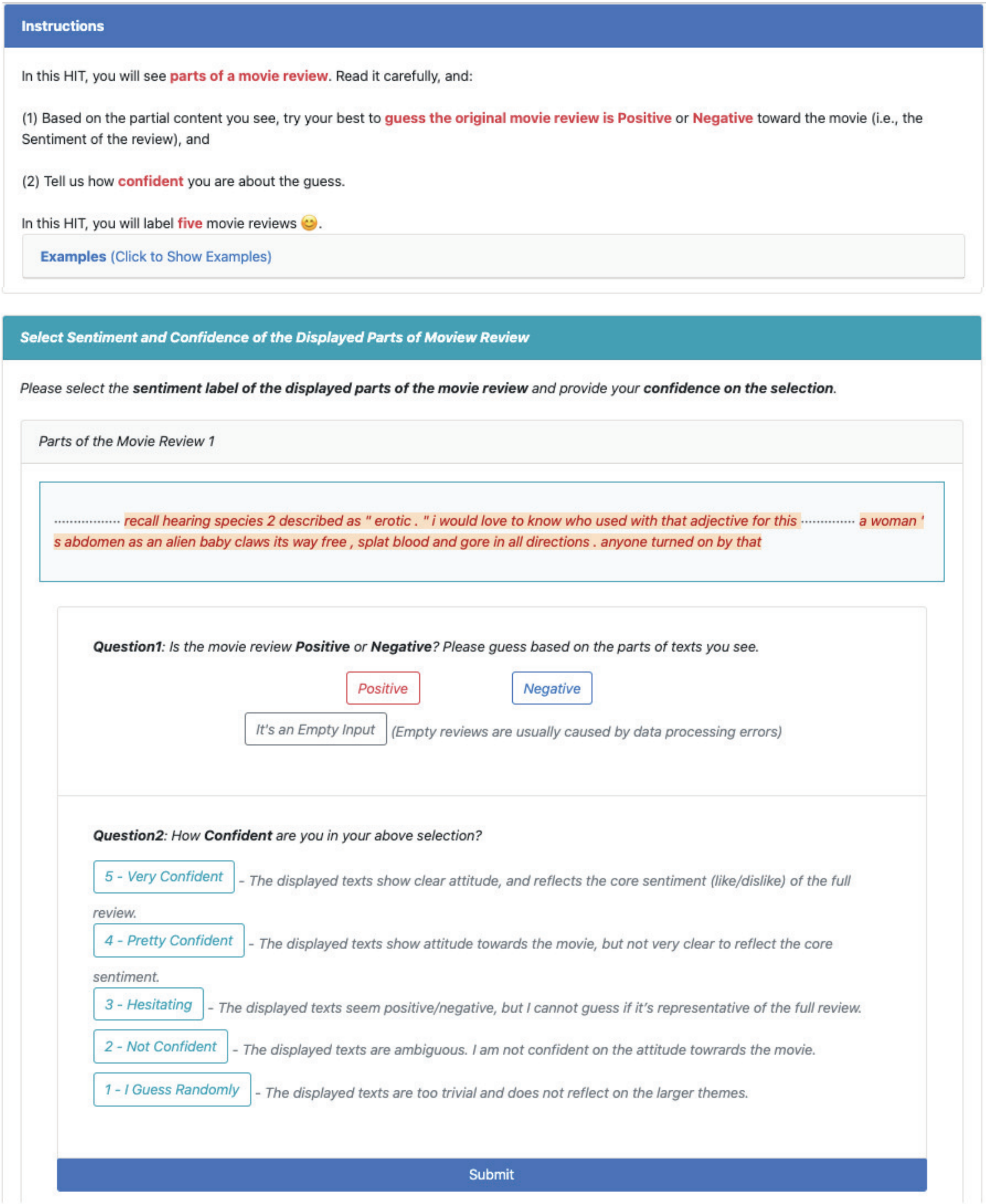
In detail, given the rationale length k, we first got the count of total tokens in rationale as #tokens = k. Next, we computed the average number of rationale segments m, which are generated by LimitedInk, over the Movie dataset. We randomly selected m spans with total tokens' count as #tokens from the full input texts, thus obtaining the random baselines. We evenly separated 10 worker groups to finish five random baseline HITs and LimitedInk HITs each. We determined that good model rationales should get higher human accuracy compared with same-length random baselines.

A.3.2 Human Evaluation User Interface

We provide our designed user interfaces used in the human study. Specifically, we show the interface of the human study panel in Figure 5 (B). We also provide the detailed instructions for workers to understand our task, the instruction inteface is shown in Figure 6.

	Review1	Review2	Review3	Review4	Review5
Worker Group 1	Our@10%	Our@20%	Our@30%	Our@40%	Our@50%
Worker Group 2	Our@20%	Our@30%	Our@40%	Our@50%	Our@10%
Worker Group 3	Our@30%	Our@40%	Our@50%	Our@10%	Our@20%
Worker Group 4	Our@40%	Our@50%	Our@10%	Our@20%	Our@30%
Worker Group 5	Our@50%	Our@10%	Our@20%	Our@30%	Our@40%
Worker Group 6	Random@10%	Random@20%	Random@30%	Random@40%	Random@50%
Worker Group 7	Random@20%	Random@30%	Random@40%	Random@50%	Random@10%
Worker Group 8	Random@30%	Random@40%	Random@50%	Random@10%	Random@20%
Worker Group 9	Random@40%	Random@50%	Random@10%	Random@20%	Random@30%
Worker Group 10	Random@50%	Random@10%	Random@20%	Random@30%	Random@40%

(A) Worker Group Assignment



Instructions

In this HIT, you will see **parts of a movie review**. Read it carefully, and:

(1) Based on the partial content you see, try your best to **guess the original movie review is Positive** or **Negative** toward the movie (i.e., the Sentiment of the review), and

(2) Tell us how **confident** you are about the guess.

In this HIT, you will label **five** movie reviews 😊.

Examples (Click to Show Examples)

Select Sentiment and Confidence of the Displayed Parts of Moview Review

*Please select the **sentiment label of the displayed parts of the movie review** and provide your **confidence on the selection**.*

Parts of the Movie Review 1

.... recall hearing species 2 described as " erotic . " i would love to know who used with that adjective for this .... a woman ' s abdomen as an alien baby claws its way free , splat blood and gore in all directions . anyone turned on by that

***Question1**: Is the movie review **Positive** or **Negative**? Please guess based on the parts of texts you see.*

(Empty reviews are usually caused by data processing errors)

***Question2**: How **Confident** are you in your above selection?*

5 - Very Confident - The displayed texts show clear attitude, and reflects the core sentiment (like/dislike) of the full review.

4 - Pretty Confident - The displayed texts show attitude towards the movie, but not very clear to reflect the core sentiment.

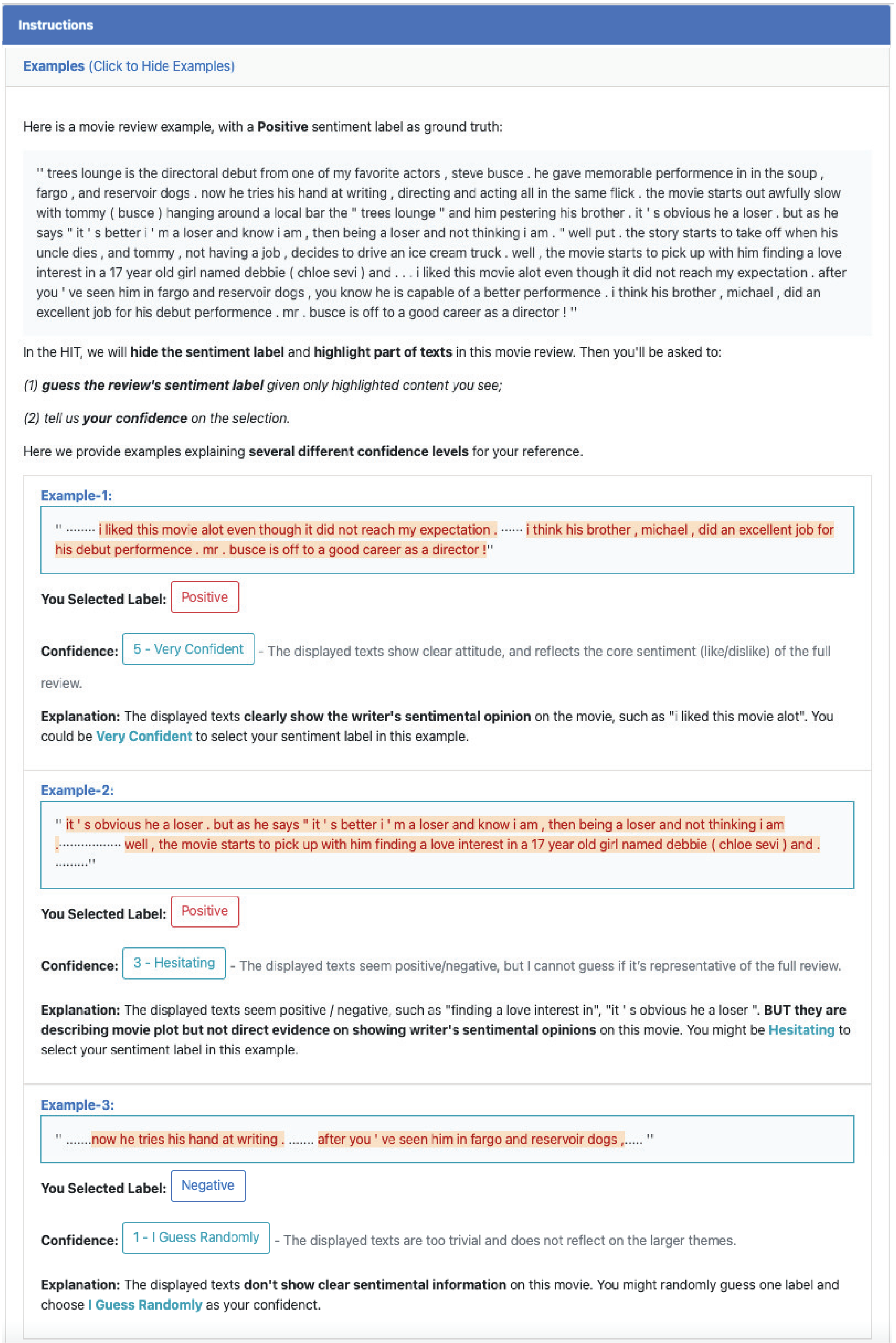
3 - Hesitating - The displayed texts seem positive/negative, but I cannot guess if it's representative of the full review.

2 - Not Confident - The displayed texts are ambiguous. I am not confident on the attitude towrards the movie.

1 - I Guess Randomly - The displayed texts are too trivial and does not reflect on the larger themes.

(B) Worker Study Interface

Figure 5: (A) The design of the worker group assignment in our human study. (B) The worker interface of the human study.



Instructions

Examples (Click to Hide Examples)

Here is a movie review example, with a **Positive** sentiment label as ground truth:

'' trees lounge is the directoral debut from one of my favorite actors , steve busce . he gave memorable performence in in the soup , fargo , and reservoir dogs . now he tries his hand at writing , directing and acting all in the same flick . the movie starts out awfully slow with tommy (busce) hanging around a local bar the " trees lounge " and him pestering his brother . it ' s obvious he a loser . but as he says " it ' s better i ' m a loser and know i am , then being a loser and not thinking i am . " well put . the story starts to take off when his uncle dies , and tommy , not having a job , decides to drive an ice cream truck . well , the movie starts to pick up with him finding a love interest in a 17 year old girl named debbie (chloe sevi) and . . . i liked this movie alot even though it did not reach my expectation . after you ' ve seen him in fargo and reservoir dogs , you know he is capable of a better performence . i think his brother , michael , did an excellent job for his debut performence . mr . busce is off to a good career as a director ! ''

In the HIT, we will **hide the sentiment label** and **highlight part of texts** in this movie review. Then you'll be asked to:

(1) ***guess the review's sentiment label*** *given only highlighted content you see;*

(2) tell us ***your confidence*** *on the selection.*

Here we provide examples explaining **several different confidence levels** for your reference.

Example-1:

'' .... i liked this movie alot even though it did not reach my expectation . ... i think his brother , michael , did an excellent job for his debut performence . mr . busce is off to a good career as a director !''

You Selected Label: Positive

Confidence: 5 - Very Confident - The displayed texts show clear attitude, and reflects the core sentiment (like/dislike) of the full review.

Explanation: The displayed texts **clearly show the writer's sentimental opinion** on the movie, such as "i liked this movie alot". You could be **Very Confident** to select your sentiment label in this example.

Example-2:

'' it ' s obvious he a loser . but as he says " it ' s better i ' m a loser and know i am , then being a loser and not thinking i am .... well , the movie starts to pick up with him finding a love interest in a 17 year old girl named debbie (chloe sevi) and . ...''

You Selected Label: Positive

Confidence: 3 - Hesitating - The displayed texts seem positive/negative, but I cannot guess if it's representative of the full review.

Explanation: The displayed texts seem positive / negative, such as "finding a love interest in", "it ' s obvious he a loser ". **BUT they are describing movie plot but not direct evidence on showing writer's sentimental opinions** on this movie. You might be **Hesitating** to select your sentiment label in this example.

Example-3:

'' ....now he tries his hand at writing . ... after you ' ve seen him in fargo and reservoir dogs ,..... ''

You Selected Label: Negative

Confidence: 1 - I Guess Randomly - The displayed texts are too trivial and does not reflect on the larger themes.

Explanation: The displayed texts **don't show clear sentimental information** on this movie. You might randomly guess one label and choose **I Guess Randomly** as your confidenct.

Figure 6: User Interface of the instruction in the human study.

Analyzing Wrap-Up Effects through an Information-Theoretic Lens

Clara Meister[👀] **Tiago Pimentel**[📖] **Thomas Hikaru Clark**[⏱]
Ryan Cotterell[👀] **Roger Levy**[⏱]

[👀]ETH Zürich [📖]University of Cambridge [⏱]Massachusetts Institute of Technology
clara.meister@inf.ethz.ch tp472@cam.ac.uk thclark@mit.edu
ryan.cotterell@inf.ethz.ch rplevy@mit.edu

Abstract

Numerous analyses of reading time (RT) data have been implemented—all in an effort to better understand the cognitive processes driving reading comprehension. However, data measured on words at the end of a sentence—or even at the end of a clause—is often omitted due to the confounding factors introduced by so-called "wrap-up effects," which manifests as a skewed distribution of RTs for these words. Consequently, the understanding of the cognitive processes that might be involved in these wrap-up effects is limited. In this work, we attempt to learn more about these processes by examining the relationship between wrap-up effects and information-theoretic quantities, such as word and context surprisals. We find that the distribution of information in prior contexts is often predictive of sentence- and clause-final RTs (while not of sentence-medial RTs). This lends support to several prior hypotheses about the processes involved in wrap-up effects.

1 Introduction

Reading puts the unfolding of linguistic input in the hands—or, really, the eyes—of the reader. Consequently, it presents a unique opportunity to gain a better understanding of how humans comprehend written language. The rate at which humans choose to read text (and process its information) should be determined by their goal of understanding it. Ergo, examining where a reader spends their time should help us to understand the nature of language comprehension processes themselves. Indeed, studies analyzing reading times have been employed to explore a number of psycholinguistic theories (e.g., Smith and Levy, 2013; Futrell et al., 2020; Van Schijndel and Linzen, 2021).

One behavior revealed by such studies is the tendency for humans to spend more time[1] on the last word of a sentence or clause. While the existence of such **wrap-up effects** is well-known (Just et al., 1982; Hill and Murray, 2000; Rayner et al., 2000; Camblin et al., 2007), the cognitive processes giving rise to them are still not fully understood. This is likely (at least in part) due to the dearth of analyses targeting naturalistic sentence-final reading behavior. First, most studies of online processing omit data from these words to explicitly control for the confounding factors wrap-up effects introduce (e.g., Smith and Levy, 2013; Goodkind and Bicknell, 2018). Second, the few studies on wrap-up effects rely on small datasets, none of which analyze naturalistic text (Just and Carpenter, 1980; Rayner et al., 2000; Kuperberg et al., 2011). This work addresses this gap, using several large corpora of reading time data. Specifically, we study whether information-theoretic concepts (such as surprisal) provide insights into the cognitive processes that occur at a sentence's boundary. Notedly, information-theoretic approaches have been proven effective for analyzing sentence-medial reading time behavior.

We follow the long line of work that has connected information-theoretic measures and psychometric data (Frank et al., 2015; Goodkind and Bicknell, 2018; Wilcox et al., 2020; Meister et al., 2021 , *inter alia*), employing similar methods to build models of sentence- and clause-final RTs. Using surprisal estimates from state-of-the-art language models, we search for a link between wrap-up effects and the information content within a sentence. We find that the distribution of surprisals of prior context is often predictive of sentence- and clause-final reading times (RTs), while not adding significant predictive power to models of sentence-medial RTs. This result suggests that the nature of cognitive processes involved during the reading of these boundary words may indeed be different than those at other positions. Such findings lend support to several prior hypotheses regarding which processes may underlie wrap-up effects

[1] Longer reading times in self-paced reading studies and longer fixation times in eye-tracking studies.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 20 - 28
May 22-27, 2022 ©2022 Association for Computational Linguistics

(e.g., the resolution of prior ambiguities), while providing evidence against other speculations (e.g., that the time spent at sentence boundaries can be quantified with a constant factor, independent of the processing difficulty of the text itself).

2 The Process of Reading

Decades of research on reading behavior have improved our understanding of the cognitive processes involved in reading comprehension (Just and Carpenter, 1980; Rayner and Clifton, 2009 , *inter alia*). Here, we will briefly describe overarching themes that are relevant for understanding wrap-up effects.

2.1 Incrementality and its Implications

It is widely accepted that language processing is incremental in nature, i.e., readers process text one word at a time (Hale, 2001, 2006; Rayner and Clifton, 2009; Boston et al., 2011 , *inter alia*). Consequently, much can be uncovered about reading comprehension via studies that analyze cognitive processing at the word-level. Many pyscholinguistic studies make use of this notion, taking per-word RTs in self-paced reading (SPR) or eye-tracking studies to be a direct reflection of the processing load of that word (e.g., Smith and Levy, 2013; Van Schijndel and Linzen, 2021). This RT–processing effort relationship then allows us to identify relationships between a word's processing load and its attributes (e.g., surprisal or length)—which in turn hints at the underlying cognitive processes involved in comprehension. One prominently studied attribute is word predictability; a notion naturally quantified by **surprisal** (also known as Shannon's (1948) information content). Formally, the surprisal of a word w is defined as $s(w) \overset{\text{def}}{=} -\log p(w \mid \mathbf{w}_{<t})$, i.e., a unit's negative log-probability given the prior sentential context $\mathbf{w}_{<t}$. Notedly, this operationalization provides a way of quantifying how our prior expectations can affect our ability to process a linguistic signal.

There are several hypothesis about the mathematical nature of the relationship between per-word surprisal and processing load.[2] While there has been much empirical proof that surprisal estimates serve as a good predictor of word-level RTs (Smith and Levy, 2013; Goodkind and Bicknell, 2018; Wilcox et al., 2020), the data observed from sentence-final words appears not to follow the same relationship. Specifically, in comparison to sentence-medial words, sentence- or clause-final words are associated with increased RTs in self-paced studies (Just et al., 1982; Hill and Murray, 2000) and both increased fixation and regression times in eye-tracking studies (Rayner et al., 2000; Camblin et al., 2007). Such behavior has also been observed in controlled settings—for example, Rayner et al. (1989) found that readers fixated longer on a word when it ended a clause than when the same word did not end a clause.

Such wide-spread experimental evidence suggests sentence-final and sentence-medial reading behaviors differ from each other, and that other cognitive processes (besides standard word-level processing) effort may be at play. Yet unfortunately, these wrap-up effects have received relatively little attention in the psycholinguistic community: Most reading time studies simply exclude sentence-final (or even clause-final) words from their analyses, claiming that the (poorly-understood) effects are confounding factors in understanding the reading process (e.g., Frank et al., 2013, 2015; Wilcox et al., 2020). Rather, we believe this data can potentially provide new insights in their own right.

2.2 Wrap-up Effects

It remains unclear what exactly occurs in the mind of the reader at the end of a sentence or clause. Which cognitive processes are encompassed by the term **wrap-up effects**? Several theories have been posited. First, Just and Carpenter (1980) hypothesize that wrap-up effects include actions such as "the constructions of inter-clause relations." Second, Rayner et al. (2000) suggest they might involve attempts to resolve previously postponed comprehension problems, which could have been deferred in the hope that upcoming words would resolve the problem. Third, Hirotani et al. (2006) posit the hesitation when crossing clause boundaries is out of efficiency (Jarvella, 1971); readers do not want to have to return to the clause later, so they take the extra time to make sure there are no inconsistencies in the prior text.

While some prior hypotheses have been largely dismissed (see Stowe et al., 2018 for a more detailed summary) due to, e.g., the wide-spread support of theories of incremental processing, most others lack formal testing in naturalistic reading studies. We attempt to address this gap.

[2] Surprisal theory (Hale, 2001), for instance, posits a linear relation.

Concretely, we posit the relationship between text's information-theoretic attributes and its observed wrap-up times can provide an indication of the presence (or lack) of several cognitive processes that are potentially a part of sentence wrap-up. For example, high-surprisal words in the preceding context may correlate with the presence of ambiguities in the text; they may also correlate with complex linguistic relationships of the current text with prior sentences—which are two driving forces in the theories given above. Consequently, in this work, we ask whether the reading behavior observed at the end of a sentence or clause can be described (at least partially) by the distribution of information content in the preceding context,[3] as this may give insights for several prior hypotheses about wrap-up effects.

3 Language Models as Predictors of Psychometric Data

Formally, a language model $\widehat{p}$ is a probability distribution over natural language sentences. In the case when $\widehat{p}$ is locally normalized, which is the predominant case for today's neural language models, $\widehat{p}$ is defined as the product of conditional probability distributions: $\widehat{p}(\mathbf{y}) = \prod_{t=1}^{|\mathbf{y}|} \widehat{p}(y_t \mid \mathbf{y}_{<t})$, where each $\widehat{p}(\cdot \mid \mathbf{y}_{<t})$ is a distribution with support over linguistic units y (typically words) from a set vocabulary $\mathcal{V}$, which includes a special end-of-sequence token. Consequently, we can use $\widehat{p}$ to estimate individual word probabilities. Model parameters are typically estimated by minimizing the negative log-likelihood of a corpus of natural language strings $\mathcal{C}$, i.e., minimizing $\mathcal{L}(\widehat{p}) = -\sum_{\mathbf{y}\in\mathcal{C}} \log \widehat{p}(\mathbf{y})$.

One widely embraced technique in information-theoretic psycholinguistics is the use of these language models to estimate the probabilities required for computing surprisal (Hale, 2001; Demberg and Keller, 2008; Mitchell et al., 2010; Fernandez Monsalve et al., 2012). It has even been observed that a language model's perplexity[4] correlates negatively with the psychometric predictive power provided by its surprisal estimates (Frank and Bod, 2011; Goodkind and Bicknell, 2018; Wilcox et al., 2020). If these language models keep improving at their current fast pace (Radford et al., 2019; Brown et al., 2020), exciting new results in computational psycholinguistics may follow, connecting reading behavior to the statistics of natural language.

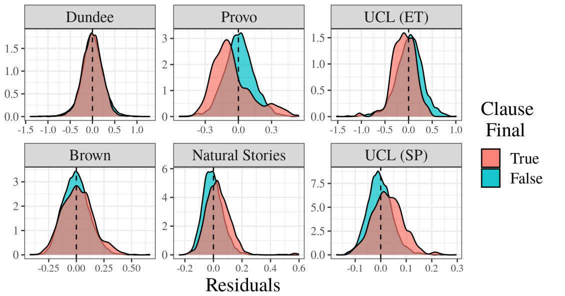


Figure 1: Distributions of residuals when predicting either clause-final or non clause-final times using our baseline linear models. Models are fit to (the log-transform of) non clause-final average RTs. Outlier times (according to log-normal distribution) are excluded. The top level datasets contain eye-tracking data while the bottom contain SPR data. Full distributions of RTs are shown in App. B, where we also show models fit to regression times, rather than full reading times.

Predicting Reading Times. In the computational psycholinguistics literature, the RT–surprisal relationship is typically studied using predictive models: RTs are predicted using surprisal estimates (along with other attributes such as number of characters) for the current word. The predictive power of these models, together with the structure of the model itself (which defines a specific relationship between RTs and surprisal), is then used as evidence of the studied effect. While this paradigm is successful in modeling sentence-medial RTs (Smith and Levy, 2013; Goodkind and Bicknell, 2018; Wilcox et al., 2020), its effectiveness for modeling sentence- and clause-final times is largely unknown due to the omission of this data from the majority of RT analyses.

A priori, we might expect per-word surprisal to be a similarly powerful predictor of sentence and clause-final RTs.[5] Yet in Fig. 1, we see that when our baseline linear model (described more precisely in §4) is fit to sentence-medial RTs, the residuals for predictions of clause-final RTs appear to be neither normally distributed nor centered around 0. Further, these trends appear to be different for eye-tracking and SPR data, where the latter are skewed towards *lower* values for all datasets.[6] These re-

[3] Importantly, the research questions we ask are not concerned with describing the *full* set of cognitive processes that occur at the end of a clause or sentence—or even whether there is a *causal* relationship between information content and sentence- and clause-final RTs.

[4] Perplexity is a monotonic function of the average surprisal of linguistic units in-context under a model.

[5] Several works (e.g., Stowe et al., 2018) have argued the cognitive processes involved in comprehension of clause-final words are exactly the same as those for sentence-medial words.

[6] The opposite is true for regression times in eye-tracking data; see App. B.

sults provide further confirmation that clause-final data does not adhere to the same relationship with RT as sentence-medial data, a phenomenon that may perhaps be accounted for by additional factors at play in the comprehension of clause-final words. Thus, we ask whether taking into account information from the entire prior context can give us a better model of these clause-final RTs.

To this end, we operationalize the information content INF in text $\mathbf{w}$ (of length T) as:[7]

$$\textsc{inf}^{(k)}(\mathbf{w}) \overset{\text{def}}{=} \sum_{t=1}^{T} s(w_t)^k \qquad (k \geq 0) \qquad (1)$$

where $\mathbf{w}$ may be an entire sentence, or only its first T words. Notably, the case of $k = 0$ returns T; under $k = 1$, we get the total information content of $\mathbf{w}$. For $k > 1$, moments of high-surprisal will disproportionately drive up the value of $\textsc{inf}^{(k)}(\mathbf{w})$. Such words may indicate, e.g., moments of ambiguity or uneven distributions of information in text. Thus, how well $\textsc{inf}^{(k)}(\mathbf{w})$ (as a function of k) predicts model sentence- and clause-final RTs may indicate which attributes of prior text (if any) can be linked to the additional cognitive processes involved in wrap-up effects.

4 Experiments

Data. We use reading time data from 5 corpora over 2 modalities: the Natural Stories (Futrell et al., 2018), Brown (Smith and Levy, 2013), and UCL (SP) (Frank et al., 2013) Corpora, which contain SPR data, as well as the Provo (Luke and Christianson, 2018), Dundee (Kennedy et al., 2003) and UCL (ET) (Frank et al., 2013) Corpora, which contain eye movements during reading. All corpora are in English. For eye-tracking data, we take reading time to be the sum over all fixation times on that word. We provide an analysis of regression (a.k.a. go-past) time in App. B. We provide further details regarding pre-processing in App. A.

Estimating Surprisal. We obtain surprisal estimates from three language models: GPT-2 (Radford et al., 2019), TransformerXL (Dai et al., 2019) and a 5-gram model, estimated using Modified Kneser–Essen–Ney Smoothing (Ney et al., 1994). We compute per-word surprisal as the sum of subword surprisals, when applicable. Additionally, punctuation is included in these estimates, although see App. B for results omitting punctuation, which are qualitatively the same. More details are given in App. A.

Evaluation. Following Wilcox et al. (2020) and Meister et al. (2021), we quantify the predictive power of a variable of interest ($\textsc{inf}^{(k)}(\mathbf{w})$ here) as the mean difference in log-likelihood ΔLogLik of a (held-out) data point when using a model with and without that predictor. In other words, we train two models to predict RTs—one with and one without access to $\textsc{inf}^{(k)}(\mathbf{w})$—the difference in their predictive power is ΔLogLik. A positive ΔLogLik value indicates the model with this predictor fits the observed data more closely than a model without this predictor. We use 10-fold cross-validation to compute ΔLogLik values so as to avoid overfitting, taking the mean across the held-out folds as our final metric. Our baseline model for predicting per-word RTs contains predictors for surprisal, unigram log-frequency, character length, and the interaction of the latter two. These values, albeit computed on the previous word, are also included to account for spill-over effects (Smith and Levy, 2013). Surprisal from two words back is included for SPR datasets. Unless otherwise stated, GPT-2 estimates are used for baseline surprisal estimates in all models.

Results. Here we explore the additional predictive power that $\textsc{inf}^{(k)}$ gives us when modeling clause-final RTs. In Fig. 2, we observe that often the additional information provided by $\textsc{inf}^{(k)}(\mathbf{w})$ indeed leads to better models of clause-final RTs. In most cases, $\textsc{inf}^{(k)}$ at some value of $k > 0$ leads to larger gains in predictive power than $k = 0$. Ergo, the information content of the preceding text is more indicative of wrap-up behavior than length alone. Further, while often within standard error, $\textsc{inf}^{(k)}(\mathbf{w})$ at $k > 1$ provides more predictive power than at $k = 1$ across the majority of datasets. This indicates that unevenness in the distribution of surprisal is stronger than the total surprisal content alone as a predictor of clause-final RTs. The same experiments for sentence-medial words show these quantities are less helpful when modeling their RTs. Note that these effects hold above and beyond the spill-over effects from the window immediately preceding the sentence boundary. The effect of the distribution of surprisal throughout the sentence is stronger for eye-tracking data than for SPR; further, the trends are even more pronounced when measuring *regression times* for eye-tracking data (see App. B).

[7] We note Meister et al. (2021) used similar operationalizations to test for evidence in support of the uniform information density hypothesis.

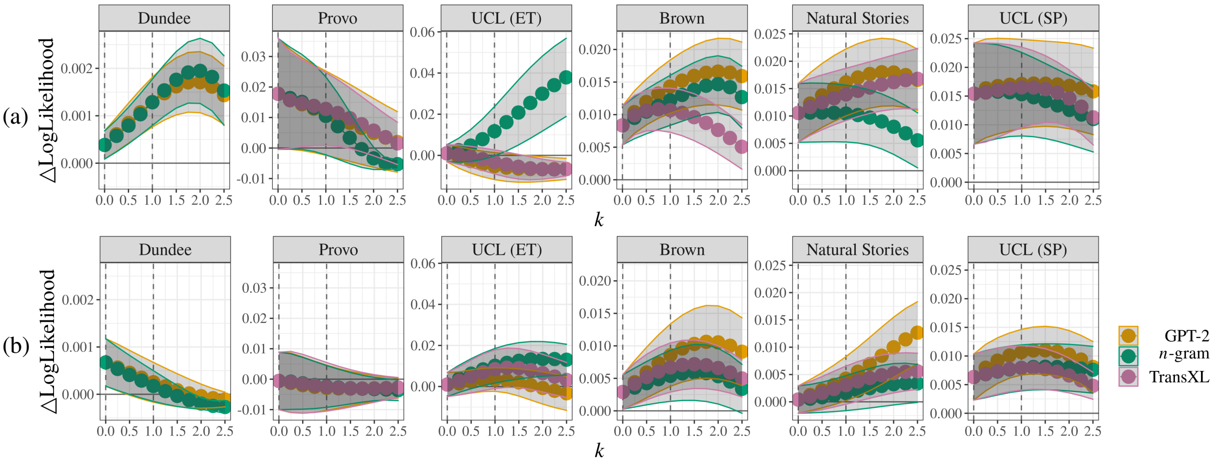


Figure 2: Mean ΔLogLik as a function of the exponent k in $\text{INF}^{(k)}$ for models of sentence and clause-final (top row) and sentence-medial (bottom row) RTs using surprisal estimates from different language models. Shaded region connects standard error estimates. Vertical intercepts at $k = 0, 1$ are for reference. We see that our information-theoretic predictors contribute much less modeling power to the prediction of sentence-medial RTs in comparison to sentence- and clause-final RTs.

Notably, we see some variation in trends across datasets. Due to the nature of psycholinguistic studies, it is natural to expect some variation due to, e.g., data collection procedures or inaccuracies from measurement devices. Another (perhaps more influential) factor in the difference in trends comes from the variation in dataset sizes. We see that with the smaller datasets (e.g., UCL and Provo), there may not be enough data to learn accurate model parameters. This artifact may manifest as the noisiness or a lack of a significant increase in log-likelihood (on a held-out test set) over the baseline that we observe in some cases.

When considering prior theories of wrap-up processes, these results have several implications. For example, they can be interpreted as supporting and extending Rayner et al.'s (2000) hypothesis, which suggests the extra time at sentence boundaries is spent resolving prior ambiguities. In this case, the observed correlation between wrap-up times and $\text{INF}^{(k)}(\mathbf{w})$ may potentially be linked to two factors: (1) contextual ambiguities increasing variation in per-word information content; and (2) contextual ambiguities being resolved at clause ends. On the other hand, these results provide evidence against the hypothesis that the cognitive processes occurring during the comprehension of sentence-medial and clause-final words are the same. Further, it also goes against Hirotani et al.'s (2006) hypothesis (discussed in §2.2), as the differences in sentence-medial and clause-final times cannot be purely quantified by a constant factor.

5 Conclusion

We attempt to shed light on the nature of wrap-up effects by exploring the relationship between clause-final RTs and information-theoretic attributes of text. We find that operationalizations of the information contained in preceding context lead to better predictions of these RTs, while not adding significant predictive power for sentence-medial RTs. This suggests that information-theoretic attributes of text can shed light on the cognitive processes happening during the comprehension of clause-final words. Further, these processes may indeed be different in nature than those required for sentence-medial words. In short, our results provide evidence (either in support or against) about several theories of the nature of wrap-up processes.

Ethics Statement

All studies involving human evaluations were conducted outside of the scope of this paper. The authors foresee no ethical concerns with the work presented in this paper.

Acknowledgments

RC acknowledges support from the Swiss National Science Foundation (SNSF) as part of the "The Forgotten Role of Inductive Bias in Interpretability" project. TP is supported by a Facebook Fellowship Award. RPL acknowledges support from NSF grant 2121074.

References

Marisa Ferrara Boston, John T. Hale, Shravan Vasishth, and Reinhold Kliegl. 2011. Parallel processing and sentence comprehension difficulty. *Language and Cognitive Processes*, 26(3):301–349.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

C. Christine Camblin, Peter C. Gordon, and Tamara Y. Swaab. 2007. The interplay of discourse congruence and lexical association during sentence processing: Evidence from ERPs and eye tracking. *Journal of Memory and Language*, 56(1):103–128.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Vera Demberg and Frank Keller. 2008. Data from eye-tracking corpora as evidence for theories of syntactic processing complexity. *Cognition*, 109(2):193–210.

Irene Fernandez Monsalve, Stefan L. Frank, and Gabriella Vigliocco. 2012. Lexical surprisal as a general predictor of reading time. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 398–408, Avignon, France. Association for Computational Linguistics.

Stefan L. Frank and Rens Bod. 2011. Insensitivity of the human sentence-processing system to hierarchical structure. *Psychological Science*, 22(6):829–834.

Stefan L. Frank, Irene Fernandez Monsalve, Robin L. Thompson, and Gabriella Vigliocco. 2013. Reading time data for evaluating broad-coverage models of English sentence processing. *Behavior Research Methods*, 45:1182–1190.

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

Richard Futrell, Edward Gibson, and Roger P. Levy. 2020. Lossy-context surprisal: An information-theoretic model of memory effects in sentence processing. *Cognitive Science*, 44:e12814.

Richard Futrell, Edward Gibson, Harry J. Tily, Idan Blank, Anastasia Vishnevetsky, Steven Piantadosi, and Evelina Fedorenko. 2018. The Natural Stories Corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*. European Language Resources Association.

Adam Goodkind and Klinton Bicknell. 2018. Predictive power of word surprisal for reading times is a linear function of language model quality. In *Proceedings of the 8th Workshop on Cognitive Modeling and Computational Linguistics*, pages 10–18.

John Hale. 2001. A probabilistic Earley parser as a psycholinguistic model. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.

John Hale. 2006. Uncertainty about the rest of the sentence. *Cognitive science*, 30(4):643–672.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland. Association for Computational Linguistics.

Robin Hill and Wayne Murray. 2000. *Commas and Spaces: Effects of Punctuation on Eye Movements and Sentence Parsing*, pages 565–590. Elsevier.

Masako Hirotani, Lyn Frazier, and Keith Rayner. 2006. Punctuation and intonation effects on clause and sentence wrap-up: Evidence from eye movements. *Journal of Memory and Language*, 54(3):425–443.

Robert J. Jarvella. 1971. Syntactic processing of connected speech. *Journal of Verbal Learning and Verbal Behavior*, 10(4):409–416.

Marcel Adam Just and Patricia A. Carpenter. 1980. A theory of reading: From eye fixations to comprehension. *Psychological Review*, 87 4:329–54.

Marcel Adam Just, Patricia A. Carpenter, and Jacqueline D. Woolley. 1982. Paradigms and processes in reading comprehension. *Journal of Experimental Psychology: General*, 111:228–238.

Alan Kennedy, Robin Hill, and Joel Pynte. 2003. The Dundee Corpus. In *Proceedings of the 12th European Conference on Eye Movements*.

Gina R. Kuperberg, Martin Paczynski, and Tali Ditman. 2011. Establishing Causal Coherence across Sentences: An ERP Study. *Journal of Cognitive Neuroscience*, 23(5):1230–1246.

Steven G. Luke and Kiel Christianson. 2018. The Provo Corpus: A large eye-tracking corpus with predictability norms. *Behavior Research Methods*, 50(2):826–833.

Clara Meister, Tiago Pimentel, Patrick Haller, Lena Jäger, Ryan Cotterell, and Roger Levy. 2021. Revisiting the uniform information density hypothesis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online. Association for Computational Linguistics.

Jeff Mitchell, Mirella Lapata, Vera Demberg, and Frank Keller. 2010. Syntactic and semantic factors in processing difficulty: An integrated measure. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 196–206, Uppsala, Sweden. Association for Computational Linguistics.

Hermann Ney, Ute Essen, and Reinhard Kneser. 1994. On structuring probabilistic dependences in stochastic language modelling. *Computer Speech and Language*, 8(1):1–38.

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

Keith Rayner and Charles Clifton. 2009. Language processing in reading and speech perception is fast and incremental: Implications for event-related potential research. *Biological Psychology*, 80(1):4–9.

Keith Rayner, Gretchen Kambe, and Susan A. Duffy. 2000. The effect of clause wrap-up on eye movements during reading. *The Quarterly Journal of Experimental Psychology Section A*, 53(4):1061–1080.

Keith Rayner, Sara C. Sereno, Robin K. Morris, A. Réne Schmauder, and Charles Clifton Jr. 1989. Eye movements and on-line language comprehension processes. *Language and Cognitive Processes*, 4(3–4):SI21–SI49.

Claude E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

Nathaniel J. Smith and Roger Levy. 2013. The effect of word predictability on reading time is logarithmic. *Cognition*, 128(3):302–319.

Laurie A. Stowe, Edith Kaan, Laura Sabourin, and Ryan C. Taylor. 2018. The sentence wrap-up dogma. *Cognition*, 176:232–247.

Marten Van Schijndel and Tal Linzen. 2021. Single-stage prediction models do not explain the magnitude of syntactic disambiguation difficulty. *Cognitive Science*, 45(6):e12988.

Ethan Gotlieb Wilcox, Jon Gauthier, Jennifer Hu, Peng Qian, and Roger Levy. 2020. On the predictive power of neural language models for human real-time comprehension behavior. In *Proceedings of the 42nd Annual Meeting of the Cognitive Science Society*, pages 1707–1713. Cognitive Science Society.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Experimental Setup

A.1 Data Pre-processing

We use the Moses decoder[8] tokenizer and punctuation normalizer to pre-process all text data. Some of the Hugging Face tokenizers for respective neural models performed additional tokenization; we refer the reader to the library documentation for more details. We determine clause-final words as all those ending in punctuation. Capitalization was kept intact albeit the lowercase version of words were used in unigram probability estimates. We estimate unigram log-probabilities on WikiText-103 using the KenLM (Heafield, 2011) library with default hyperparameters. We removed outlier word-level reading times (specifically those with a z-score > 3 when the distribution was modeled as log-linear).

A.2 Surprisal Estimates

We use pre-trained neural language models to compute most surprisal estimates. For reproducibility, we employ the model checkpoints provided by Hugging Face (Wolf et al., 2020). Specifically, for GPT-2, we use the default OpenAI version (`gpt2`); for TransformerXL, we use a version of the model (architecture described in Dai et al. (2019)) that has been fine-tuned on WikiText-103 (`transfo-xl-wt103`); for BERT, we use the `bert-base-cased` version. Notably, BERT models the probability of a word given both prior and *later* context, which means it can only give us pseudo estimates of surprisal. Both GPT-2 and BERT use sub-word tokenization. We additionally use surprisal estimates from a 5-gram model trained on WikiText-103 using the KenLM (Heafield, 2011) library with default hyperparameters for Kneser–Essen–Ney smoothing.

B Additional Results

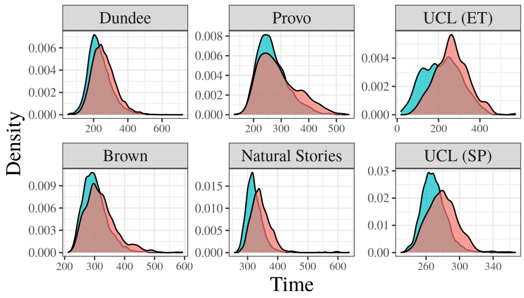


Figure 3: Distributions of average RTs for clause-final and non-clause-final words. Outlier times (according to log-normal distribution) are excluded from averages for both graphs. The top level datasets contain eye-tracking data while the bottom contain SPR data.

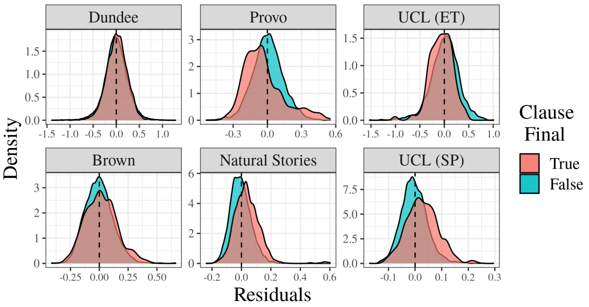
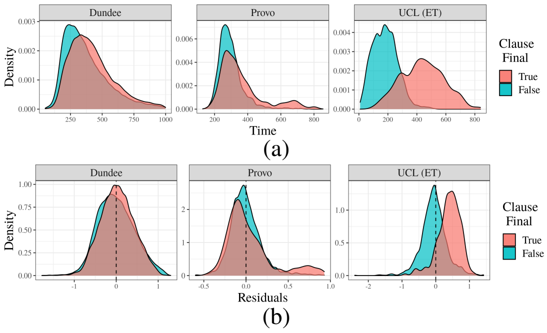


Figure 4: Version of Fig. 1 where surprisal estimates do *not* include the surprisal assigned to punctuation, which is often a large contributor to clause-final surprisal estimates. We see very little qualitative difference with Fig. 1.

B.1 Regression Times Analysis

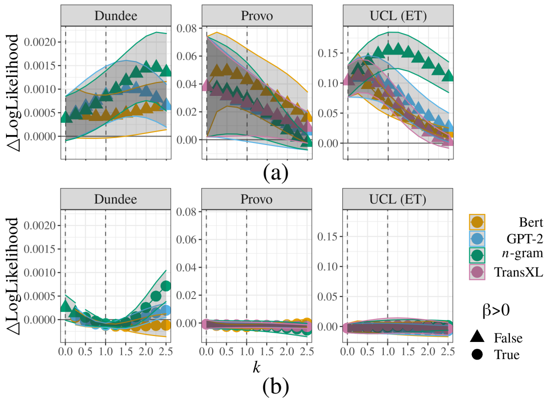


(a)

(b)

Figure 5: Version of (a) Fig. 3 and (b) Fig. 1 for regression times for clause-final and non-clause-final words. Only applicable for eye-tracking datasets

[8] `http://www.statmt.org/moses/`



(b)

Figure 6: Same setup as Fig. 2 albeit predicting regression times. Only applicable for eye-tracking datasets. (a) shows results for predicting clause-final words, while (b) shows results for predicting sentence-medial words.

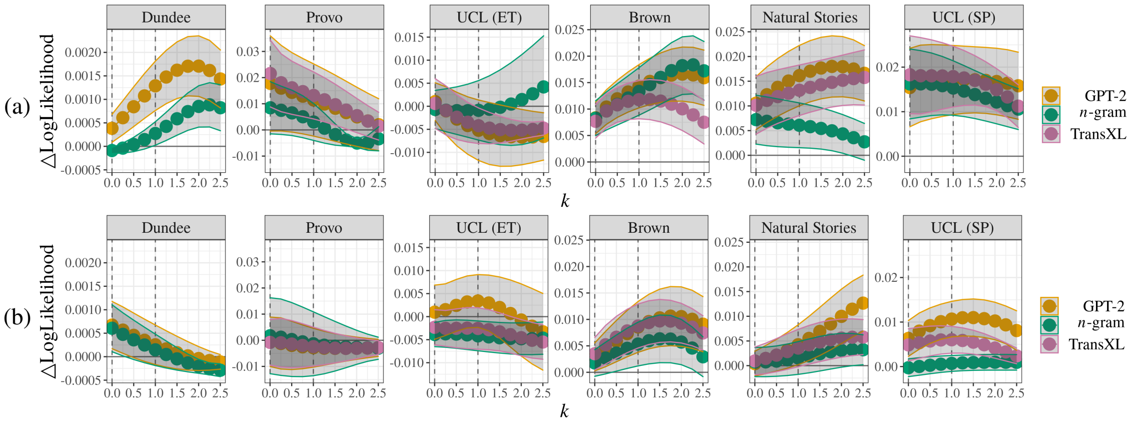


Figure 7: Same setup as Fig. 2 albeit using respective model estimates for the baseline per-word surprisal estimate. (a) shows results for predicting clause-final words, while (b) shows results for predicting sentence-medial words. Results follow similar trends to those seen in Fig. 2.

Have my arguments been replied to? Argument Pair Extraction as Machine Reading Comprehension

Jianzhu Bao[1,2*], **Jingyi Sun**[1,2*], **Qinglin Zhu**[1,2], **Ruifeng Xu**[1,3†]
[1]Harbin Institute of Technology (Shenzhen), China
[2]Joint Lab of China Merchants Securities and HITSZ
[3]Peng Cheng Laboratory, Shenzhen, China
{jianzhubao, sunjingyihit}@gmail.com
zhuqinglin@stu.hit.edu.cn, xuruifeng@hit.edu.cn

Abstract

Argument pair extraction (APE) aims to automatically mine argument pairs from two interrelated argumentative documents. Existing studies typically identify argument pairs indirectly by predicting sentence-level relations between two documents, neglecting the modeling of the holistic argument-level interactions. Towards this issue, we propose to address APE via a machine reading comprehension (MRC) framework with two phases. The first phase employs an argument mining (AM) query to identify all arguments in two documents. The second phase considers each identified argument as an APE query to extract its paired arguments from another document, allowing to better capture the argument-level interactions. Also, this framework enables these two phases to be jointly trained in a single MRC model, thereby maximizing the mutual benefits of them. Experimental results demonstrate that our approach achieves the best performance, outperforming the state-of-the-art method by 7.11% in F_1 score.

1 Introduction

As a salient part of argument mining (AM), the analysis of dialogical argumentation has received increasing research attention (Morio and Fujita, 2018; Chakrabarty et al., 2019; Ji et al., 2021; Cheng et al., 2021; Yuan et al., 2021). Argument pair extraction (APE), proposed by Cheng et al. (2020), is a new task within this field that focuses on extracting interactive argument pairs from two interrelated documents (e.g., peer reviewer and rebuttal). Figure 1 presents an example of APE where two interrelated documents are segmented into arguments and non-arguments at sentence level. Two arguments from different documents that discuss the same issues are regarded as an argument pair.

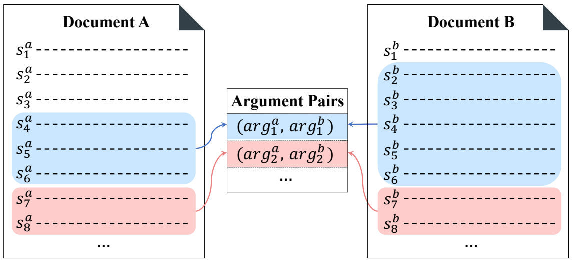


Figure 1: A simplified example of APE task, where each dashed line in the two documents denotes a sentence. s_j^i is the j-th sentence in document i, and arg_j^i is an argument in the j-th argument pair from document i. Sentences without colors indicate non-arguments, while sentences covered by colors can form arguments. Two arguments with the same color are regarded as an argument pair.

Previous works (Cheng et al., 2020, 2021) commonly address APE by decomposing it into two sentence-level subtasks, i.e., a sequence labeling task and a sentence relation classification task. These methods identify arguments by sentence-level sequence labeling and determine whether two sentences belong to the same argument pair by sentence relation classification. Afterwards, the argument pairs are inferred indirectly by certain rules combining the results of the two subtasks. However, such a paradigm only considers sentence-level relations, while the holistic argument-level relations can not be well modeled.

In this paper, we argue that APE can be considered as a multi-turn machine reading comprehension (MRC) task with two phases, i.e., an AM phase and an APE phase. Specifically, in the first turn, a special AM query is employed to identify all the arguments in the first document (AM phase). Afterwards, in each subsequent turn, every identified argument is treated as an APE query to extract its paired arguments from the second document (APE phase). Similarly, this process can also be performed in another direction, that is, using the

[*] Equal Contribution
[†] Corresponding Author

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 29 - 35
May 22-27, 2022 ©2022 Association for Computational Linguistics

arguments identified in the second document as queries to extract the paired arguments from the first document. We train these two phases jointly in a single MRC model, allowing them to benefit each other. By considering arguments as queries, our proposed MRC framework can better capture the interactions between each query argument and the queried document, thus extracting the argument pairs at the argument level. In addition, considering the long length of the documents, we utilize Longformer (Beltagy et al., 2020) to model longer contexts.

We evaluate our method on the large benchmark dataset (Cheng et al., 2020). Results show that our proposed method significantly outperforms the current state-of-the-art method by 7.11% in F_1 score.

2 Related Work

2.1 Argument Mining

Argument mining aims to analyze the structure of argumentation, and it contains various subtasks, such as argument component identification (Moens et al., 2007; Goudas et al., 2015; Ajjour et al., 2017; Jo et al., 2019), argument relation prediction (Nguyen and Litman, 2016; Cocarascu et al., 2020; Jo et al., 2021), argumentation structure parsing (Stab and Gurevych, 2017; Kuribayashi et al., 2019; Morio et al., 2020; Bao et al., 2021), argumentation strategy analysis (Khatib et al., 2018; Morio et al., 2019), etc.

Most previous works mainly focus on monological argumentation, while dialogical argumentation (Morio and Fujita, 2018; Chakrabarty et al., 2019) is relatively less emphasized. Recently, the analysis of dialogical argumentation has attracted increasing attention in the field of argument mining. Cheng et al. (2020) propose the APE task which involves identifying arguments and extracting argument pairs in peer review and rebuttal. Ji et al. (2021) identify interactive argument pairs in online debate forums based on the discrete variational autoencoders. Cheng et al. (2021) address the APE task based on a table-filling approach. Yuan et al. (2021) construct a dialogical argumentation knowledge graph for identifying argument pairs.

2.2 Machine Reading Comprehension

Machine reading comprehension (MRC) aims to extract answer spans from a passage according to a given query (Seo et al., 2017; Chen et al., 2017; Devlin et al., 2019; Wen et al., 2021). Formulating NLP tasks as MRC tasks has been a rising trend in recent years, such as dependency parsing (Gan et al., 2021), relation extraction (Levy et al., 2017), named entity recognition (Li et al., 2020), sentiment analysis (Chen et al., 2021; Mao et al., 2021). Unlike previous studies above, we employ a MRC framework to analyze the complex argumentative relations between two documents with excessively long length.

3 Methodology

3.1 Task Formulation

We assume that two interrelated documents $D_a = (s_1^a, s_2^a, ..., s_{n^a}^a)$ and $D_b = (s_1^b, s_2^b, ..., s_{n^b}^b)$ are given, where s_j^i denotes the j-th sentence in document i. We need to extract the collection of argument pairs $P = \{(arg_i^a, arg_i^b)\}_{i=1}^{|P|}$, where arg_i^a and arg_i^b respectively represent the arguments in document D_a and D_b, and they compose the i-th argument pair. Note that each argument consists of one or more consecutive sentences. For example, $arg_i^a = (s_{start}^{a,i}, s_{start+1}^{a,i}, ..., s_{end}^{a,i})$ where $start$ and end denote the start and end sentence index.

To frame APE as a multi-turn MRC task, two types of queries are constructed, i.e., the argument mining (AM) query and the argument pair extraction (APE) query. Intuitively, we could consider the process of extracting argument pairs from the perspective of two directions, i.e., $D_a \rightarrow D_b$ and $D_b \rightarrow D_a$. For the $D_a \rightarrow D_b$ direction, we first construct an AM query using a special token whose corresponding answers are all the arguments in document D_a. After recognizing all arguments through the AM query, each recognized argument is considered as an APE query whose corresponding answers are its paired arguments in document D_b. Similarly, for the $D_b \rightarrow D_a$ direction, we first query document D_b with the AM query, and then generate the APE queries for document D_a. Finally, the argument pairs can be derived by fusing the answer results of all APE queries.

3.2 MRC Framework

3.2.1 Encoder

Since APE is a document-level task with excessively long text, we adopt Longformer to capture contextual information over longer distances. For brevity, we only describe the MRC process in the $D_a \rightarrow D_b$ direction below, and the $D_b \rightarrow D_a$ direction can be performed similarly.

Formally, we use a special token "[AM]" to represent the AM query q^{am}, which aims to identify all the arguments $A^a = \{arg_k^a\}_{k=1}^{|A^a|}$ in document D_a where arg_k^a indicates the k-th argument in D_a. Then, each identified argument arg_k^a is considered as an APE query $q_k^{a,ape}$, i.e., $q_k^{a,ape} = arg_k^a = (s_{start}^{a,k}, ..., s_{end}^{a,k})$. Note that we use gold arguments as APE queries during training.

With these queries, we first concatenate the AM query q^{am} and the document D_a as an input sequence for AM:

$$I^{am} = ([\text{s}], q^{am}, [/\text{s}], [\text{s}], s_1^a, s_2^a, ..., s_{n^a}^a, [/\text{s}]) \tag{1}$$

Also, we concatenate each APE query $q_k^{a,ape}$ and the document D_b to obtain multiple input sequences for APE:

$$I_k^{ape} = ([\text{s}], q_k^{a,ape}, [/\text{s}], [\text{s}], s_1^b, s_2^b, ..., s_{n^b}^b, [/\text{s}]) \tag{2}$$

where [s] and [/s] are special tokens of Longformer.

Subsequently, for each sequence above, we feed it into Longformer to get the hidden representation of each token in the input document. Specifically, to enable Longformer to better learn argument-specific representations, we add global attention to the tokens of the query. Afterwards, we derive the hidden representation of each sentence through mean pooling on token representations in this sentence. Further, to better model the long-term dependency among sentences, the hidden representations of sentences are fed into LSTM to derive the contextual sentence representation matrix $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n)$.

3.2.2 Answer Span Prediction

For each turn, one or more answer spans will be extracted as arguments. Note that, in each direction, the first turn aims to extract all arguments, while the following turns aim to extract arguments that can form pairs with the query argument.

Specifically, inspired by Li et al. (2020), we fed $\mathbf{H}$ into two binary classifiers to predict the start and end sentence positions of arguments. After obtaining all start and end positions, we further employ another binary classifier to determine whether each start and end position pair (matched by Cartesian product) forms an answer span. Note that the input of this span classifier is the concatenation of the start and end sentence representations from $\mathbf{H}$.

3.2.3 Training

During training, the three classifiers described in Section 3.2.2 yield three cross-entropy losses, i.e., a start loss, an end loss, and a span loss. We simply sum these losses up as the training objective of our model. In addition, the AM phrase and the APE phrase are trained jointly in a single MRC model.

3.2.4 Inference

During inference, the $D_a \rightarrow D_b$ direction uses the trained MRC model to first identify all the arguments in D_a by the AM query and then extract all the argument pairs in D_b by the APE queries. Similarly, the $D_b \rightarrow D_a$ direction can be performed in the same manner by simply exchanging the order of D_a and D_b. Each APE query in both directions yields one or more argument pairs, where each argument pair contains the query argument and one extracted argument. We simply merge all argument pairs extracted by all APE queries into a union set to obtain the final inference results.

4 Experiments

4.1 Experimental setup

4.1.1 Dataset

Our experiments are conducted on the large APE benchmark dataset, namely the Review-Rebuttal (RR) dataset (Cheng et al., 2020), which contains 4,764 pairs of review-rebuttal passages of ICLR. Following the setup of (Cheng et al., 2021), we also evaluate our method on two versions of the train/dev/test (8:1:1) split, i.e., RR-Passage-v1 and RR-Submission-v2. Note that in our method, we view review passage and rebuttal passage as document D_a and document D_b, respectively.

4.1.2 Implementation Details

We adopt Longformer-base-4096 [1] as base encoder, and we use sliding window attention with the window size of 512. We train our model 6 epochs with a batch size of 4. AdamW (Kingma and Ba, 2015) is used as the optimizer, and the learning rates for Longformer and other layers are 1e-5 and 1e-3.[2]

The evaluation metrics contain two aspects, namely AM and APE. Different from (Cheng et al., 2021, 2020), sentence pairing is not included as a metric because we extract argument pairs directly.

[1] https://huggingface.co/allenai/longformer-base-4096

[2] Our source code is available at https://github.com/HLT-HITSZ/MRC_APE

Data	Methods	Argument Mining			Argument Pair Extraction		
		Pre.	Rec.	F_1	Pre.	Rec.	F_1
RR-Submission-v2	PL-H-LSTM-CRF	67.02	68.49	67.75	19.74	19.13	19.43
	MT-H-LSTM-CRF	70.74	69.46	70.09	27.24	26.00	26.61
	MLMC	69.53	**73.27**	71.35	37.15	29.38	32.81
	MRC-APE-Bert	**73.36**	68.35	70.77	42.26	34.06	37.72
	MRC-APE-Sep.	72.45	71.58	72.01	41.09	36.99	38.93
	MRC-APE (Ours)	71.83	73.05	**72.43**	**41.83**	**38.17**	**39.92**
RR-Passage-v1	PL-H-LSTM-CRF	73.10	67.65	70.27	21.24	19.30	20.22
	MT-H-LSTM-CRF	71.85	71.01	71.43	30.08	29.55	29.81
	MLMC	66.79	**72.17**	69.38	**40.27**	29.53	34.07
	MRC-APE-Bert	66.81	69.84	68.29	34.70	35.53	35.11
	MRC-APE-Sep.	75.27	67.90	71.39	36.63	40.05	38.26
	MRC-APE (Ours)	**76.39**	70.62	**73.39**	37.70	**44.00**	**40.61**

Table 1: Main results on RR-Submission-v2 and RR-Passage-v1 (%). The best scores are in bold.

We select the best parameters based on the performance (i.e., average F_1 scores of AM and APE) on the dev set. All scores are averaged across 5 distinct trials using different random seeds.

4.1.3 Baselines

We compare our model with several baselines. **PL-H-LSTM-CRF** (Cheng et al., 2020) independently trains an argument mining task and a sentence pairing task, while **MT-H-LSTM-CRF** (Cheng et al., 2020) trains two subtasks in a multi-task framework. **MLMC** (Cheng et al., 2021) is an attention-guided model based on a table-filling approach, which is the current state-of-the-art method.

Furthermore, we implement two additional baselines. For a fair comparison with MLMC, **MRC-APE-Bert** replaces Longformer with Bert, where documents with excessively long length are splited into several segments. Instead of jointly training AM and APE phases, **MRC-APE-Sep.** trains the two phases separately.

4.2 Results and Analysis

4.2.1 Main Results

As shown in Table 1, our model achieves the best performance on both versions of the RR dataset. Concretely, on RR-Submission-v2, our model significantly outperforms the current state-of-the-art model MLMC by at least 7.11% in APE F_1 score. On RR-Passage-v1, our model obtains at least a 6.54% higher APE F_1 score than the MLMC. Also, our model achieves the best performance on AM. Furthermore, without applying Longformer as the base encoder, MRC-APE-Bert still outperforms MLMC in APE F_1 score, demonstrating that our improvement is not only brought by Longformer. However, for the AM task, MAC-APE-Bert achieves slightly lower F_1 score than MLMC. The reason may be that, in MLMC, the predictions of the AM task are influenced by the APE task through a complex attention interaction mechanism. However, our model does not require such a complex design and can achieve much better results on the APE task. Besides, our MRC-APE achieves better results than MRC-APE-Sep. on both AM and APE tasks, indicating that jointly training two phases in a single MRC model could maximize the mutual benefits of the two phases.

Method	APE			$\Delta(\mathbf{F}_1)$
	Pre.	Rec.	F_1	
MRC-APE (Ours)	41.83	**38.17**	**39.92**	-
w/o $D_b \rightarrow D_a$	**49.47**	31.33	38.36	1.56
w/o $D_a \rightarrow D_b$	46.68	26.02	33.41	6.51
w/o LSTM	44.98	34.51	39.06	0.86
w/o GA	38.20	30.66	34.02	5.90

Table 2: The results of ablation experiments on RR-Submission-v2 (%). The best scores are in bold. *w/o* GA indicates that the global attention is not included in Longformer.

In addition, to analyze the error propagation from the first phase to the second phase, we use the true label of AM task to predict APE task. Under this setting, our model can achieve around 59.44% F_1 score for APE task, showing effectiveness in identifying argument pairs.

4.2.2 Ablation Study

The ablation study results are shown in Table 2. It can be observed that using two directions contributes greatly to our method. Also, using the arguments recognized in D_a to extract the paired arguments in D_b is more critical in the RR dataset, removing it causes a 6.51% decrease in APE F_1 score. Without the LSTM to capture the long-term dependency among sentences, the APE F_1 score decreases by 0.86%. Furthermore, the performance drops heavily without the global attention, because it enables more interactions between the query argument and the queried document, thus better argument-specific representations could be learned.

5 Conclusion

In this paper, we propose to frame the argument pair extraction (APE) task as a machine reading comprehension (MRC) task. Our MRC framework addresses APE through two phases with two types of queries, that is, argument mining (AM) query and argument pair extraction (APE) query. Our proposed method can better model the argument-level interactions, thus facilitating the extraction of argument pairs. Experimental results on a large benchmark dataset demonstrate that our proposed method achieves state-of-the-art performance.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (61876053, 62006062, 62176076), the Shenzhen Foundational Research Funding (JCYJ20200109113441941, JCYJ20210324115614039), Joint Lab of HITSZ and China Merchants Securities.

References

Yamen Ajjour, Wei-Fan Chen, Johannes Kiesel, Henning Wachsmuth, and Benno Stein. 2017. Unit segmentation of argumentative texts. In *Proceedings of the 4th Workshop on Argument Mining, ArgMining@EMNLP 2017, Copenhagen, Denmark, September 8, 2017*, pages 118–128. Association for Computational Linguistics.

Jianzhu Bao, Chuang Fan, Jipeng Wu, Yixue Dang, Jiachen Du, and Ruifeng Xu. 2021. A neural transition-based model for argumentation mining. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6354–6364. Association for Computational Linguistics.

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

Tuhin Chakrabarty, Christopher Hidey, Smaranda Muresan, Kathy McKeown, and Alyssa Hwang. 2019. AMPERSAND: argument mining for persuasive online discussions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2933–2943. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1870–1879. Association for Computational Linguistics.

Shaowei Chen, Yu Wang, Jie Liu, and Yuelin Wang. 2021. Bidirectional machine reading comprehension for aspect sentiment triplet extraction. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12666–12674. AAAI Press.

Liying Cheng, Lidong Bing, Qian Yu, Wei Lu, and Luo Si. 2020. APE: argument pair extraction from peer review and rebuttal via multi-task learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7000–7011. Association for Computational Linguistics.

Liying Cheng, Tianyu Wu, Lidong Bing, and Luo Si. 2021. Argument pair extraction via attention-guided multi-layer multi-cross encoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6341–6353. Association for Computational Linguistics.

Oana Cocarascu, Elena Cabrio, Serena Villata, and Francesca Toni. 2020. Dataset independent baselines for relation prediction in argument mining. In *Computational Models of Argument - Proceedings of COMMA 2020, Perugia, Italy, September 4-11, 2020*, volume 326 of *Frontiers in Artificial Intelligence and Applications*, pages 45–52. IOS Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Leilei Gan, Yuxian Meng, Kun Kuang, Xiaofei Sun, Chun Fan, Fei Wu, and Jiwei Li. 2021. Dependency parsing as mrc-based span-span prediction. *CoRR*, abs/2105.07654.

Theodosios Goudas, Christos Louizos, Georgios Petasis, and Vangelis Karkaletsis. 2015. Argument extraction from news, blogs, and the social web. *Int. J. Artif. Intell. Tools*, 24(5):1540024:1–1540024:22.

Lu Ji, Zhongyu Wei, Jing Li, Qi Zhang, and Xuanjing Huang. 2021. Discrete argument representation learning for interactive argument pair identification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5467–5478. Association for Computational Linguistics.

Yohan Jo, Seojin Bang, Chris Reed, and Eduard H. Hovy. 2021. Classifying argumentative relations using logical mechanisms and argumentation schemes. *Trans. Assoc. Comput. Linguistics*, 9:721–739.

Yohan Jo, Jacky Visser, Chris Reed, and Eduard H. Hovy. 2019. A cascade model for proposition extraction in argumentation. In *Proceedings of the 6th Workshop on Argument Mining, ArgMining@ACL 2019, Florence, Italy, August 1, 2019*, pages 11–24. Association for Computational Linguistics.

Khalid Al Khatib, Henning Wachsmuth, Kevin Lang, Jakob Herpel, Matthias Hagen, and Benno Stein. 2018. Modeling deliberative argumentation strategies on wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2545–2555. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Tatsuki Kuribayashi, Hiroki Ouchi, Naoya Inoue, Paul Reisert, Toshinori Miyoshi, Jun Suzuki, and Kentaro Inui. 2019. An empirical study of span representations in argumentation structure parsing. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4691–4698. Association for Computational Linguistics.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017*, pages 333–342. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5849–5859. Association for Computational Linguistics.

Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. A joint training dual-mrc framework for aspect based sentiment analysis. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13543–13551. AAAI Press.

Marie-Francine Moens, Erik Boiy, Raquel Mochales Palau, and Chris Reed. 2007. Automatic detection of arguments in legal texts. In *The Eleventh International Conference on Artificial Intelligence and Law, Proceedings of the Conference, June 4-8, 2007, Stanford Law School, Stanford, California, USA*, pages 225–230. ACM.

Gaku Morio, Ryo Egawa, and Katsuhide Fujita. 2019. Revealing and predicting online persuasion strategy with elementary units. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6273–6278. Association for Computational Linguistics.

Gaku Morio and Katsuhide Fujita. 2018. End-to-end argument mining for discussion threads based on parallel constrained pointer architecture. In *Proceedings of the 5th Workshop on Argument Mining, ArgMining@EMNLP 2018, Brussels, Belgium, November 1, 2018*, pages 11–21. Association for Computational Linguistics.

Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, Yuta Koreeda, and Kohsuke Yanai. 2020. Towards better non-tree argument mining: Proposition-level biaffine parsing with task-specific parameterization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 3259–3266. Association for Computational Linguistics.

Huy Nguyen and Diane J. Litman. 2016. Context-aware argumentative relation mining. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Christian Stab and Iryna Gurevych. 2017. Parsing argumentation structures in persuasive essays. *Comput. Linguistics*, 43(3):619–659.

Haoyang Wen, Anthony Ferritto, Heng Ji, Radu Florian, and Avi Sil. 2021. VAULT: variable unified long text representation for machine reading comprehension. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 1035–1042. Association for Computational Linguistics.

Jian Yuan, Zhongyu Wei, Donghua Zhao, Qi Zhang, and Changjian Jiang. 2021. Leveraging argumentation knowledge graph for interactive argument pair identification. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2310–2319. Association for Computational Linguistics.

On the probability–quality paradox in language generation

Clara Meister🤯 **Gian Wiher**🤯 **Tiago Pimentel**🥱 **Ryan Cotterell**🤯
🤯ETH Zürich 🥱University of Cambridge
clara.meister@inf.ethz.ch gian.wiher@inf.ethz.ch
tp472@cam.ac.uk ryan.cotterell@inf.ethz.ch

Abstract

When generating natural language from neural probabilistic models, high probability does not always coincide with high quality: It has often been observed that mode-seeking decoding methods, i.e., those that produce high-probability text under the model, lead to unnatural language. On the other hand, the lower-probability text generated by stochastic methods is perceived as more human-like. In this note, we offer an explanation for this phenomenon by analyzing language generation through an information-theoretic lens. Specifically, we posit that human-like language should contain an amount of information (quantified as negative log-probability) that is close to the entropy of the distribution over natural strings. Further, we posit that language with substantially more (or less) information is undesirable. We provide preliminary empirical evidence in favor of this hypothesis; quality ratings of both human and machine-generated text—covering multiple tasks and common decoding strategies—suggest high-quality text has an information content significantly closer to the entropy than we would expect by chance.

1 Introduction

Today's probabilistic neural language models are often trained on millions—if not billions—of lines of human text; thus, at least at an intuitive level, we would expect high-probability generations to be human-like. Yet the high-quality[1] texts these models have become famous for producing (Brown et al., 2020; Clark et al., 2021) are usually not those assigned the highest probability by the model (Fan et al., 2018; Holtzman et al., 2020; Basu et al., 2021; DeLucia et al., 2021). Rather, the relationship between probability and quality appears to have an inflection point,[2] i.e., quality and probability are positively correlated only until a certain threshold, after which the correlation becomes negative. While the existence of such a trend has received informal explanations (see, e.g., Ippolito et al. (2019) and Zhang et al. (2021) for a qualitative discussion about the trade-off between diversity and quality), it lacks a more fundamental understanding. Why does the lower probability text produced by stochastic decoding methods—such as nucleus or top-k sampling—outperform text generated using probability-maximizing approaches? In this note, we take an information-theoretic approach in an attempt to answer this question.

In information theory, probability has another interpretation: its negative log quantifies **information content**. In the context of natural language, the notion of information content is intuitive; humans use strings as a means to convey information. Further, less predictable text, i.e., text which would be harder for us to anticipate, conveys *more* information. If we assume that the goal of human communication is to transmit messages efficiently and reliably (Gibson et al., 2019), we may predict that these strings' information content should concentrate inside a specific interval. At one extreme, strings with more-than-expected information may be hard to process, and thus ought to be disfavored when producing language.[3] At the other extreme, low-information strings may be seen as boring and uninformative.

Collectively, these concepts lead us to propose the **expected information hypothesis**: Text perceived as human-like should have an information content within a small interval around the expected information—i.e., the entropy—of natural language strings. Such a hypothesis offers

[1] We assume that "human-like" is a (necessary but not sufficient) prerequisite for "high-quality" in the context of natural language strings.

[2] The inflection point is empirically demonstrated in our App. B or in Fig. 1 of Zhang et al. (2021).

[3] Many works in psycholinguistics have shown a direct relationship between information content and processing effort (Smith and Levy, 2013; Wilcox et al., 2020, *inter alia*).

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 36 - 45
May 22-27, 2022 ©2022 Association for Computational Linguistics

an intuitive explanation for the trends observed in natural language generation (NLG), i.e., why desirable text seems to exist not always at the high end of the probability spectrum but around a certain inflection point.[4] Moreover, it also gives us a *testable* hypothesis: given a language generation model q whose entropy we can empirically estimate, we can evaluate whether high-quality text indeed has an information content that falls within an interval around this quantity.

To test our hypothesis, we perform an analysis comparing human and model-generated text, investigating multiple common decoding strategies and NLG tasks. Specifically, our analysis focuses exclusively on English text. We indeed observe that the information content of highly ranked text (as judged by humans) often falls within a standard deviation of model entropy; there is statistically significant evidence that this is not due to chance. Further, the best-performing decoding methods appear to select strings with an information content within this interval. We take these observations as empirical support for our hypothesis, helping to explain the probability–quality paradox observed in language generation.

2 Probabilistic Language Generators

In this work, we focus on probabilistic models for language generation tasks. Formally, these models are probability distributions q over natural language strings $\mathbf{y} \in \mathcal{Y}$, where $\mathcal{Y}$ is the (countably infinite) set consisting of all possible strings that can be constructed from a set vocabulary $\mathcal{V}$:

$$\mathcal{Y} \overset{\text{def}}{=} \{\text{BOS} \circ \mathbf{v} \circ \text{EOS} \mid \mathbf{v} \in \mathcal{V}^*\} \tag{1}$$

Here, BOS and EOS stand for special reserved beginning- and end-of-string tokens, respectively, and $\mathcal{V}^*$ denotes the Kleene closure of $\mathcal{V}$. In practice, we limit the set of strings we consider to $\mathcal{Y}_N \subset \mathcal{Y}$ for some maximum sequence length N.

Note that q may be a conditional model. For instance, we may model $q(\cdot \mid \mathbf{x})$ where $\mathbf{x}$ is an input text, as in the case of machine translation, or an input image, as in the case of image captioning. However, for notational brevity, we omit this explicit dependence in most of our subsequent analyses. In order to estimate q, it is standard practice to maximize the log-probability of a training corpus $\mathcal{C}$ under the model with respect to the model's parameters $\boldsymbol{\theta}$. This is equivalent to minimizing its negative log-probability:

$$L(\boldsymbol{\theta}; \mathcal{C}) = -\sum_{\mathbf{y} \in \mathcal{C}} \log q(\mathbf{y}) \tag{2}$$

There are many different decision rules one can employ for generating natural language strings from a model q; such sets of rules are generally referred to as decoding strategies; see Wiher et al. (2022) for an in-depth review. Given the probabilistic nature of the models we consider, an intuitive strategy for decoding would be to choose the string with the highest probability under q, an approach referred to as maximum-a-posteriori (MAP) decoding.[5] Yet recent research has shown that solutions to MAP decoding—or, even more generally, to heuristic mode-seeking methods such as beam search—are often not high-quality, even in state-of-the-art NLG models. For example, in the domain of machine translation, the most probable string under the model is often the empty string (Stahlberg and Byrne, 2019). Similarly, in the domain of open-ended generation, mode-seeking methods produce dull and generic text (Holtzman et al., 2020).

Where maximization has failed, authors have turned to stochastic methods, taking random samples from q. While the resulting text is often assigned much lower probability than the mode, it can be qualitatively much better. This peculiarity has puzzled the language generation community for the last few years, with only qualitative intuitions being offered as explanation. This paper in turn offers a quantitative explanation.

3 Language as Communication

While many aspects of natural language may not perfectly adhere to Shannon's mathematical theory of communication, there are several characteristics of human language that *can* fruitfully be described using an information-theoretic framework.[6] Here we employ this framework for explaining recent phenomena observed in probabilistic NLG.

[4] Similar ideas have been used to improve language models and language generation before (Meister et al., 2020; Wei et al., 2021).

[5] Note that MAP decoding is somewhat of a misnomer since we are not maximizing over a Bayesian posterior. Nonetheless, the term has become commonplace in the language generation literature.

[6] A large body of work has explored the extent to which attributes of human languages—such as word lengths or phoneme distributions—can be explained as information-theoretic design features (Gibson et al., 2019). Surprisal theory, for instance, directly relates human language processing difficulty to information content (Hale, 2001).

3.1 Measuring Information

We can precisely compute the information content of a string given the *true* (perhaps conditional) probability distribution p over natural language strings. Fortunately, this is the exact distribution our language generation models in §2 are trained to approximate.[7] Assuming q approximates p well (as quantified by metrics such as perplexity), we may thus use it to estimate such attributes of natural language strings. In this work, we will measure the amount of information a specific realization $\mathbf{y}$ contains, which we denote $\mathrm{I}(\mathbf{y}) \stackrel{\text{def}}{=} -\log q(\mathbf{y})$, as well as the *expected* amount of information a random $\mathbf{y} \in \mathcal{Y}_N$ drawn from q contains, also termed the entropy of q:

$$\mathbb{E}_q\left[\mathrm{I}(\mathbf{y})\right] = \mathrm{H}(q) = -\sum_{\mathbf{y}\in\mathcal{Y}_N} q(\mathbf{y}) \log q(\mathbf{y}) \quad (3)$$

Note that Pimentel et al. (2021b, Theorem 2) prove that, as long as the probability of EOS under q is bounded below by some $\epsilon > 0$, then the entropy of q is finite. In our case we restrict q to a finite subset $\mathcal{Y}_N$ of $\mathcal{Y}$, which also implies that Eq. (3) is finite.

3.2 The Expected Information Hypothesis

Language is used as a means for transferring information. This property of language has in fact motivated several theories of language evolution; many have posited, for instance, that natural language has developed to optimize for reliable and efficient data communication, subject to cognitive resources (Zipf, 1949; Hockett, 1960; Hawkins, 2004; Piantadosi et al., 2011). The above theories arguably imply that humans tend to produce natural language strings with a certain amount of information; they also imply that, on the receiving end of communication, humans would expect similar strings. We argue that this amount is intuitively close to the language's entropy, i.e., close to the average string's information content.

Expected Information Hypothesis. *Text perceived as human-like typically encodes an amount of information close to the expected information content of natural language strings, i.e., in the interval* $[\mathrm{H}(p) - \varepsilon,\ \mathrm{H}(p) + \varepsilon]$ *for a natural language string distribution* p *and some* ε.[8] *Text that falls outside of this region is likely perceived as unnatural.*

This viewpoint can be applied to the problem of decoding neural text generators. In the context of a model q of the distribution p, this implies that—when q is a good approximation—human-like text should typically have a negative log-probability close to the entropy of q. In §4, we provide empirical evidence for this hypothesis.

Relationship to the typical set. The set of strings that we discuss has an intuitive relationship to the typical set (Shannon, 1948), an information-theoretic concept defined for stationary ergodic stochastic processes. However, generation from standard neural probabilistic language models cannot be framed as such a process.[9] While we cannot utilize the formal mathematical underpinnings of typicality, the connection can still be useful for understanding why strings with a given information content exhibit certain characteristics. An overview of the concept is in App. A for the interested reader; also see Dieleman (2020) for further insights on typicality in the context of generative models.

4 Experiments

Our experiments present an analysis of the distribution of information content in text generated by both humans and probabilistic models. Specifically, we look at the relationship between information content and quality—as measured by human judgments. We perform experiments on two natural language generation tasks: abstractive summarization and story generation. We present the results for story generation here, while the results for summarization can be found in App. B due to space constraints. A recreation of the probability versus quality plots of Zhang et al. (2021) can also be found in App. B.

We use the following Monte Carlo estimator for the entropy, i.e., expected information content, of

[7] To see this, recall that minimizing the objective in Eq. (2) is (up to an additive constant) equivalent to minimizing the Kullback–Leibler divergence—an information-theoretic quantity that measures the amount of information lost when approximating one probability distribution with another—between the empirical distribution p and our model q.

[8] While we do not offer a concrete explanation of why distributions over natural language strings have a particular entropy, we posit that it is determined by cognitive constraints, as observed with other phenomena in natural language (Coupé et al., 2019; Pimentel et al., 2021a).

[9] Specifically, most neural language models are neither stationary (due to their ability to encode arbitrarily long sequences; Welleck et al. 2020) nor ergodic (because of the absorbing nature of the EOS state). This implies that we cannot guarantee the existence of an entropy rate, which is necessary to define the typical set.

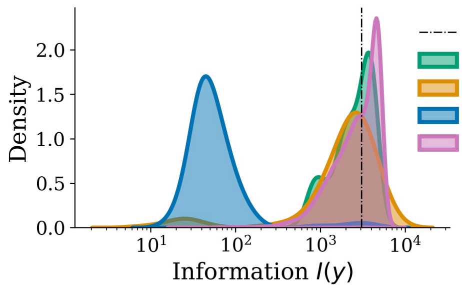


Figure 1: The distribution over information $\text{I}(\mathbf{y})$ values of: MODEL, the model, as estimated using samples from q; REFERENCE, the reference strings; TOP 1 and BOTTOM 1, model-generated strings ranked first and last (respectively) among all decoding strategies by human annotators. The latter 3 are all w.r.t. a held-out test set. Same graph is reproduced for individual decoding strategies in App. B.

our model q:

$$\widehat{\text{H}}(q) = \frac{1}{M}\sum_{m=1}^{M} -\log q(\mathbf{y}^{(m)}) \tag{4}$$

where we sample $\mathbf{y}^{(m)} \overset{\text{i.i.d.}}{\sim} q$. Algorithmically, taking these samples may be done in linear time using ancestral sampling. All computations are performed with the test sets of respective datasets. Note that for both abstractive summarization and story generation, where we condition on some input $\mathbf{x}$, we must compute the *conditional* entropy for each input, i.e., using $q(\cdot \mid \mathbf{x})$ instead of $q(\cdot)$. For each $\mathbf{x}$, we take $M = 100$ to estimate $\widehat{\text{H}}(q(\cdot \mid \mathbf{x}))$.

4.1 Setup

Models and Data. We only conduct experiments on the English language. For story generation, we fine-tune GPT-2 (medium) (Radford et al., 2019) (checkpoint made available by OpenAI) on the WritingPrompts dataset (Fan et al., 2018). For abstractive summarization, we use BART (Lewis et al., 2020), fine-tuned on the CNN/DailyMail dataset (Nallapati et al., 2016). We rely on the open-sourced code-base from the HuggingFace framework (Wolf et al., 2020) for reproducibility.

Decoding Strategies. We explore text generated according to a number of different decoding strategies. Unless otherwise stated, we use the implementation provided by Hugging Face for each of the decoding algorithms. Along with standard ancestral sampling, we experiment with the following six decoding strategies:

- **greedy search**;
- **beam search** with beam sizes $k = 5$ and $k = 10$;
- **diverse beam search** (Vijayakumar et al., 2016) with Hamming distance as a dissimilarity function and $\lambda = 0.7$ and $G = k = 5$;[10]
- **ancestral sampling**;
- **top-k sampling** (Fan et al., 2018) with $k = 30$;
- **nucleus sampling** (Holtzman et al., 2020) with $p = 0.85$;[11]
- **minimum Bayes risk decoding** (MBR; Eikema and Aziz 2020)[12] with 32 Monte Carlo samples[13] from q and BEER (Stanojević and Sima'an, 2014) as the utility function.

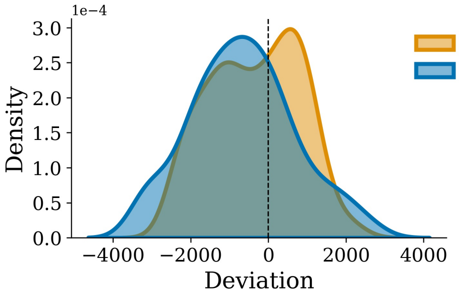


Figure 2: The distribution of the difference in total information content for (1) test-set references and (2) top-ranked model-generated strings from the (conditional) entropy of the model from which they were generated.

Human Evaluations. We use the *prolific* platform to obtain human judgments of text quality (according to 2 criteria per task) from 5 different annotators on 200 examples per decoding strategy–per task. This gives us a total of > 3000 annotated examples. We largely follow the guidelines recommended by van der Lee et al. (2021) in setting up our evaluations: For abstractive summarization, we ask annotators to rate *quality* and *accuracy* while for story generation, annotators rate *fluency* and *naturalness*. More details on our setup can be found in App. B.1.

4.2 Results

In Fig. 1, we plot the distribution of information content assigned by q to four different sets of strings: our reference (human-generated) text, the

[10] The choice of dissimilarity function and hyperparameters (λ, G, k) is based on the recommendations from the original work.

[11] This choice is based on experiments in (DeLucia et al., 2021) that suggest a parameter range $p \in [0.7, 0.9]$.

[12] We use the github.com/Roxot/mbr-nmt framework.

[13] The number of Monte Carlo samples was chosen based on the batch size constraint.

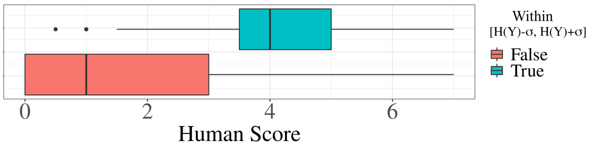


Figure 3: Human scores for strings (including both reference text and model-generated text) within 1 std of model entropy and outside of this interval. There is a statistically significant difference in means ($p < 0.001$).

top and bottom ranked (according to human annotators) strings generated from q via our different decoding strategies,[14] and strings sampled i.i.d. from q. Note that the latter should represent the distribution of negative log-probabilities assigned to strings by the model. We see that both the references and the top-ranked model-generated strings—both of which we assume are of relatively high quality—contain an amount of information clustered around the (estimated) model entropy. On the other hand, the distribution of the information content of poorly rated strings is skewed towards much lower values. The same trends hold when looking at information normalized by string length, i.e., $\text{I}(\mathbf{y})/|\mathbf{y}|$ (see App. B), demonstrating these trends are not purely an artifact of string length. We note that in our human evaluations, the reference string was ranked first in 47% of cases and it was tied for first in an additional 16% of the cases. This suggests that the quality of the reference strings is on par with—if not higher than—the set of "top 1" model-generated strings.

Fig. 2 shows the distribution of deviations of strings' information content from the model entropy;[15] results are shown for both reference strings and top-ranked model-generated strings. Because these values are distributed quite evenly around 0, we take this as additional evidence that high-quality text usually has information content close to $\text{H}(q)$. Further, the shapes of these curves motivate us to perform our next set of tests using $\varepsilon = \sigma$, the standard deviation of information values under q.[16]

We employ statistical hypothesis testing to see if the percentage of high-quality strings whose information content falls in the interval $[\text{H}(q) - \sigma, \text{H}(q) + \sigma]$ is greater than chance. For each input $\mathbf{x}$ (i.e., either a story prompt or article), we compute the information content of the reference and top-3 human-ranked strings. We then compute the percentage of items (among these four) that fall within $[\text{H}(q(\cdot \mid \mathbf{x})) - \sigma, \text{H}(q(\cdot \mid \mathbf{x})) + \sigma]$. We compare this percentage to the percentage of strings sampled directly from $q(\cdot \mid \mathbf{x})$ that falls within this interval. The former should (in expectation) be greater than the latter if the probability of high-quality strings having information content within this interval is greater than chance. Specifically, we test this using a paired, unequal-variance t-test, where samples with the same input are paired. At significance level $\alpha = 0.01$, we reject our null hypothesis—i.e., we reject that the percentage of highly rated strings (reference plus top-3 human-ranked strings) that fall within this interval is equal to (or less than) what we should expect by chance. Further, using a simple unpaired t-test, we find that the mean human score of strings (across all decoding strategies) within this region is significantly higher than those outside of this region. This characteristic is visualized in Fig. 3, where we plot the distributions of human quality ratings for strings inside and outside of this interval. We include a version of Fig. 3 further broken down by whether strings fall *above* or *below* this interval in App. B.

Additional plots reinforcing these observations can be found in App. B. Also see Meister et al. (2022) for follow-up experiments to this work.

5 Conclusion

In this work, we present the **expected information hypothesis**, which states that human-like strings typically have negative log-probability close to the expected information content of the probabilistic model from which they were generated. We use this hypothesis to explain why high-quality text seems to exist not necessarily at the high end of the probability spectrum but, rather, close to the entropy of the model. We provide empirical evidence in support of our hypothesis in an analysis of both human and machine-generated text, demonstrating that, overwhelmingly, high-quality text indeed has information content in the proposed region.

Ethics Statement

In order to complete our human evaluation, we used a crowdsourcing platform. For each task, we

[14] Specifically, for each input, we generate a single string according to each decoding strategy. We then rank these strings according to scores from human annotators.

[15] Note that this is not simply Fig. 1 shifted by a constant, as deviations are computed w.r.t. input-dependent conditional entropy estimates, i.e., $\widehat{\text{H}}(q(\cdot \mid \mathbf{x}))$.

[16] Similarly to our estimation of $\text{H}(q)$ in Eq. (3), σ can be estimated from the distribution of values of $\text{I}(\mathbf{y})$ sampled from the model.

estimated the amount of time we expected the task to take and made sure that the crowdworkers would be paid (at minimum) a wage of $15 per hour. A further ethical consideration of this work is in the context of the use of language models for text generation. Language models have been used for the generation of malicious text, e.g., fake news and triggering content. The results in this work may provide insights for those using language models for such purposes as to how generations can be chosen to seem more "human-like."

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful recommendations, as well as Sander Dieleman for feedback on a preliminary draft of this paper.

References

Sourya Basu, Govardana Sachitanandam Ramachandran, Nitish Shirish Keskar, and Lav R. Varshney. 2021. Mirostat: A perplexity-controlled neural text decoding algorithm. In *9th International Conference on Learning Representations*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, Online. Association for Computational Linguistics.

Christophe Coupé, Yoon Mi Oh, Dan Dediu, and François Pellegrino. 2019. Different languages, similar encoding efficiency: Comparable information rates across the human communicative niche. *Science Advances*, 5(9):eaaw2594.

Alexandra DeLucia, Aaron Mueller, Xiang Lisa Li, and João Sedoc. 2021. Decoding methods for neural narrative generation. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics*, pages 166–185, Online. Association for Computational Linguistics.

Sander Dieleman. 2020. Musings on typicality.

Bryan Eikema and Wilker Aziz. 2020. Is MAP decoding all you need? The inadequacy of the mode in neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4506–4520. International Committee on Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Edward Gibson, Richard Futrell, Steven T. Piantadosi, Isabelle Dautriche, Kyle Mahowald, Leon Bergen, and Roger Levy. 2019. How efficiency shapes human language. *Trends in Cognitive Sciences*.

John T. Hale. 2001. A probabilistic Earley parser as a psycholinguistic model. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics and Language Technologies*, pages 1–8.

John A. Hawkins. 2004. *Efficiency and Complexity in Grammars*. Oxford University Press, Oxford.

Charles F. Hockett. 1960. The origin of speech. *Scientific American*, 203(3):88–97.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Daphne Ippolito, Reno Kriz, João Sedoc, Maria Kustikova, and Chris Callison-Burch. 2019. Comparison of Diverse Decoding Methods from Conditional Language Models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3752–3762, Florence, Italy. Association for Computational Linguistics.

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

Clara Meister, Ryan Cotterell, and Tim Vieira. 2020. If beam search is the answer, what was the question? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 2173–2185, Online. Association for Computational Linguistics.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2022. Typical decoding for natural language generation. *CoRR*, abs/2202.00666.

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

Steven T. Piantadosi, Harry Tily, and Edward Gibson. 2011. Word lengths are optimized for efficient communication. *Proceedings of the National Academy of Sciences*, 108(9):3526–3529.

Tiago Pimentel, Clara Meister, Elizabeth Salesky, Simone Teufel, Damián Blasi, and Ryan Cotterell. 2021a. A surprisal–duration trade-off across and within the world's languages. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 949–962, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tiago Pimentel, Irene Nikkarinen, Kyle Mahowald, Ryan Cotterell, and Damián Blasi. 2021b. How (non-)optimal is the lexicon? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4426–4438, Online. Association for Computational Linguistics.

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

Claude E. Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal*, 27:623–656.

Nathaniel J. Smith and Roger Levy. 2013. The effect of word predictability on reading time is logarithmic. *Cognition*, 128(3):302–319.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.

Miloš Stanojević and Khalil Sima'an. 2014. Fitting sentence level translation evaluation with many dense features. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 202–206, Doha, Qatar. Association for Computational Linguistics.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, and Emiel Krahmer. 2021. Human evaluation of automatically generated text: Current trends and best practice guidelines. *Computer Speech & Language*, 67:101151.

Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David J. Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *CoRR*, abs/1610.02424.

Jason Wei, Clara Meister, and Ryan Cotterell. 2021. A cognitive regularizer for language modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5191–5202, Online. Association for Computational Linguistics.

Sean Welleck, Ilia Kulikov, Jaedeok Kim, Richard Yuanzhe Pang, and Kyunghyun Cho. 2020. Consistency of a recurrent language model with respect to incomplete decoding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 5553–5568, Online. Association for Computational Linguistics.

Gian Wiher, Clara Meister, and Ryan Cotterell. 2022. On decoding strategies for neural text generators. *CoRR*, abs/2203.15721.

Ethan Gotlieb Wilcox, Jon Gauthier, Jennifer Hu, Peng Qian, and Roger Levy. 2020. On the predictive power of neural language models for human real-time comprehension behavior. In *Proceedings of the Cognitive Science Society*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Hugh Zhang, Daniel Duckworth, Daphne Ippolito, and Arvind Neelakantan. 2021. Trading off diversity and quality in natural language generation. In *Proceedings of the Workshop on Human Evaluation of NLP Systems*, pages 25–33, Online. Association for Computational Linguistics.

George Kingsley Zipf. 1949. *Human Behavior and the Principle of Least Effort*. Addison-Wesley Press, Oxford, UK.

A The Typical Set

Let us imagine flipping N biased coins; specifically, let $X \sim p$ be an indicator random variable that takes values H and T. Take $p(X = \mathsf{H}) = 0.6$ and $p(X = \mathsf{T}) = 0.4$. Flipping N biased coins is then equivalent to taking N i.i.d. samples $x_n \sim p$. For reasonably large N, what might you expect the sequence $x_1, \ldots, x_N$ to look like? Few people would answer "all heads," even though this is technically the highest probability sequence. Rather, intuition tells you: an expected sequence would be one comprised of approximately 60% heads and 40% tails.

The samples that fall into the latter category have a distinctive characteristic: they contain a near-average amount of information w.r.t the support of the distribution over $X_1, \ldots, X_N$, where the information content of a realization $x_1, \ldots, x_N$ is defined as its negative log-probability. More formally, the (weakly) (ε, N)**-typical set** $A_\varepsilon^{(N)}$ for a chosen $\varepsilon > 0$ is the set of assignments $x_1, \ldots, x_N$ to random variables $\overrightarrow{X} = X_1, \ldots, X_N$ such that

$$2^{-N(\mathrm{H}(p)+\varepsilon)} \leqslant p(x_1, \ldots, x_N) \leqslant 2^{-N(\mathrm{H}(p)-\varepsilon)}$$

where $\mathrm{H}(p) \overset{\text{def}}{=} -\sum_x p(x) \log p(x)$ is the entropy—or equivalently, the expected value of the information content—of the random variable X. Under this definition we can prove that, for every $\varepsilon > 0$, there exists an N_0 such that for all $N > N_0$, we have that the (ε, N)-typical set contains at least $(1 - \varepsilon)$ of the probability mass of the joint distribution over $\overrightarrow{X}$. The concept of the typical set also generalizes to stochastic processes when we can actually compute their average information rate—or equivalently, their entropy rate.

B Experimental Design

B.1 Human Evaluations

For story generation and abstractive summarization, the raters are first presented with a news article/prompt. Next, they are presented, in random order, with the corresponding reference and the summaries/stories generated by different decoders. For each of two rating criteria, a score from 0 to 7 is assigned. For story generation the criteria are FLUENCY and NATURALNESS while for abstractive summarization QUALITY and ACCURACY are used. We provide the following short descriptions of the criteria to the raters:

FLUENCY: How fluent is the English text?

NATURALNESS: Does the text seem to be natural English text?

QUALITY: How high is the overall quality of the text?

ACCURACY: How well does the summary summarize the article?

After we obtain the ratings, we reject ratings that have not been filled out with care. Specifically, a rater is rejected if he assigns high scores to multiple examples that do not fulfill the specified criteria at all. If a rater has been rejected, we obtain a fresh set of ratings from a new rater.

C Additional Figures

We provide several additional results, looking further into the relationship between text information content and perceived quality. We see that in general, the distribution of information content of reference strings is quite close to that of the model. While the distribution of information content of top 1 ranked strings is also closer to the model distribution than many of the individual decoding strategies, the overlap is not as high as for reference strings.

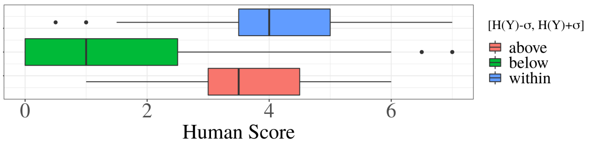


Figure 4: Human scores for strings (including both reference text and model-generated text) within 1 std of model entropy and above/below this interval. Note that "above" corresponds to text that has *lower* probability than the specified interval; due to the nature of the decoding strategies explored in this work, which all to some extent (except for ancestral sampling) disproportionately favor higher probability strings, only $< 5\%$ of all strings evaluated fall into the "above" category. Thus, we do not have a representative evaluation of this region of the probability space. However, it is often observed that extremely low-probability strings are usually incoherent or nonsensical.

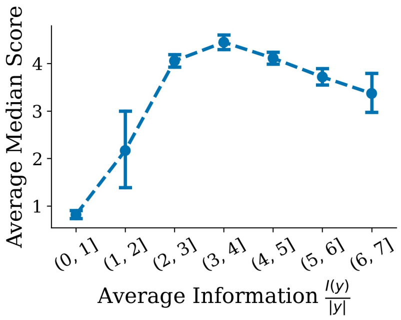


Figure 5: For story generation, median human scores (averaged across the two criterion) versus information, grouped by intervals; bars represent std. We normalize $\mathrm{I}(\mathbf{y})$ by length to mimic setup of Zhang et al. (2021), which controls for length during generation. As with Zhang et al. (2021), we see an inflection point in the relationship along the information (equivalently, negative log-probability) axis.

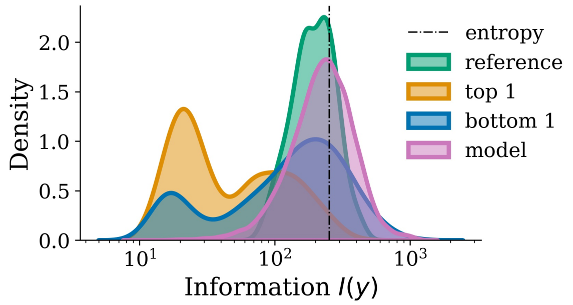


Figure 6: For abstractive summarization, the distribution over information $\mathrm{I}(\mathbf{y})$ values of: (model) the model, as estimated using samples from q; (reference) the reference strings; model-generated strings ranked (top 1) first and (bottom 1) last among all decoding strategies by human annotators. The latter 3 are all w.r.t. a held-out test set.

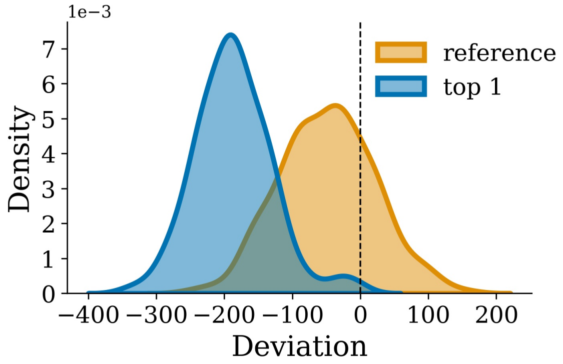


Figure 7: For abstractive summarization, the distribution of the difference in total information content for (1) test-set references and (2) top-ranked model-generated strings from the entropy of the model from which they were generated.

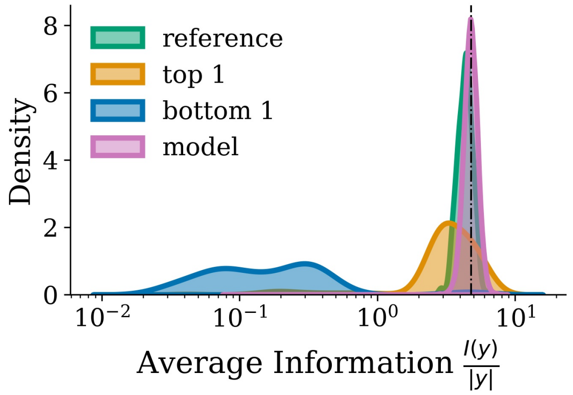
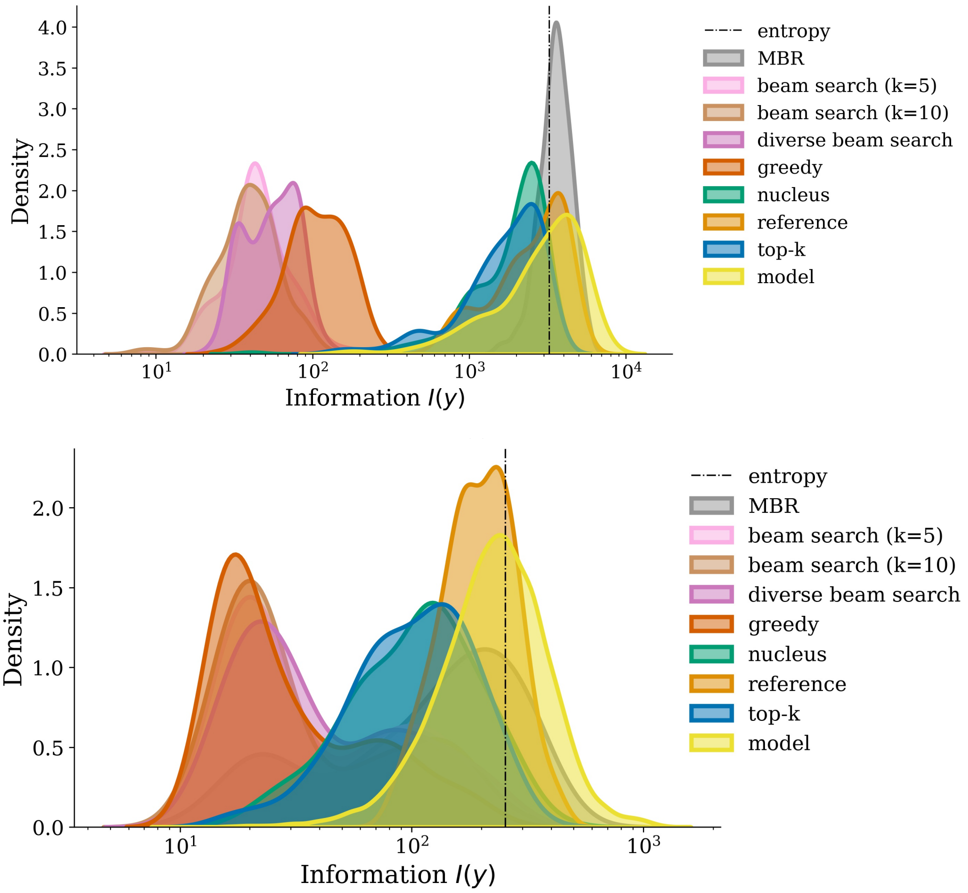


Figure 8: For story generation, the distribution over information ($\mathrm{I}(\mathbf{y})$) values normalized by length of: (model) the model, as estimated using samples from q; (reference) the reference strings; model-generated strings ranked (top 1) first and (bottom 1) last among all decoding strategies by human annotators. The latter 3 are all w.r.t. a held-out test set.



(b) a

Figure 9: The distribution over information ($\mathrm{I}(\mathbf{y})$) values for strings generated under different decoding strategies for story generation (top) and abstractive summarization (bottom). Inputs are taken from a held-out test set.

Disentangled Knowledge Transfer for OOD Intent Discovery with Unified Contrastive Learning

Yutao Mou[1*], Keqing He[2*], Yanan Wu[1*], Zhiyuan Zeng[1]
Hong Xu[1], Huixing Jiang[2], Wei Wu[2], Weiran Xu[1*]
[1]Pattern Recognition & Intelligent System Laboratory
[1]Beijing University of Posts and Telecommunications, Beijing, China
[2]Meituan Group, Beijing, China
`{myt,yanan.wu,zengzhiyuan,xuhong,xuweiran}@bupt.edu.cn`
`{hekeqing,jianghuixing,wuwei30}@meituan.com`

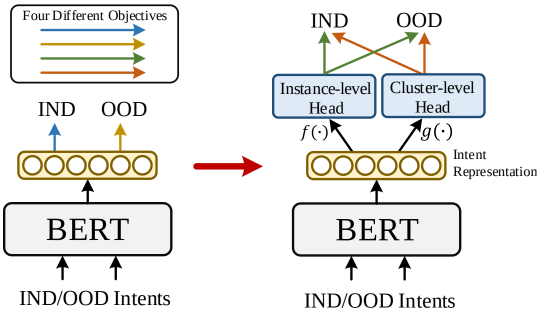
Abstract

Discovering Out-of-Domain(OOD) intents is essential for developing new skills in a task-oriented dialogue system. The key challenge is how to transfer prior IND knowledge to OOD clustering. Different from existing work based on shared intent representation, we propose a novel disentangled knowledge transfer method via a unified multi-head contrastive learning framework. We aim to bridge the gap between IND pre-training and OOD clustering. Experiments and analysis on two benchmark datasets show the effectiveness of our method. [1]

1 Introduction

Out-of-domain (OOD) intent discovery aims to group new unknown intents into different clusters, which helps improve the dialogue system for future development. Compared to existing text clustering tasks, OOD discovery considers how to leverage the prior knowledge of known in-domain (IND) intents to enhance discovering unknown OOD intents, which makes it challenging to directly apply existing clustering algorithms (MacQueen, 1967; Xie et al., 2016; Chang et al., 2017; Caron et al., 2018) to the OOD discovery task.

Previous unsupervised OOD discovery models (Hakkani-Tür et al., 2015; Padmasundari and Bangalore, 2018; Shi et al., 2018) only model OOD data but ignore prior knowledge of in-domain data thus suffer from poor performance. Therefore, recent work (Lin et al., 2020; Zhang et al., 2021) focus more on the semi-supervised setting where they firstly pre-train an in-domain intent classifier then perform clustering algorithms on extracted OOD intent representations by the pre-trained IND intent classifier. For example, Lin et al. (2020) firstly pre-trains a BERT-based (Devlin et al., 2019) IND intent classifier then uses intent representations to perform a pairwise clustering algorithm (Chang et al., 2017). Further, Zhang et al. (2021) proposes an iterative clustering method, DeepAligned, to obtain pseudo supervised signals using K-means (MacQueen, 1967). However, all of these methods ignore the matching between IND pre-training stage and OOD clustering stage because they formulate IND pre-training as the classification task while OOD clustering as the text clustering task. The different learning objectives make it hard to transfer prior IND knowledge to OOD. Besides, previous work only transfer a single intent representation from the pre-trained IND classifier to OOD clustering. Considering the entanglement of the intent representation, simply transferring IND features may harm OOD clustering. For example, there exist two levels of intent features, instance-level and class-level knowledge in the pre-trained IND classifier. Decoupling different levels of intent features helps better knowledge transferability.



a) Baseline b) Ours

Figure 1: Comparison between baselines and our proposed DKT model.

To solve the issues, we propose a novel **D**isentangled **K**nowledge **T**ransfer method (**DKT**) via a unified multi-head contrastive learning framework to transfer disentangled IND intent representations to OOD clustering. The main intuition is how to perform better knowledge transfer. As shown in Fig 1, we decouple the pre-trained intent representations into two independent subspaces, instance-level and class(cluster)-level using a uni-

[*] The first three authors contribute equally. Weiran Xu is the corresponding author.

[1] We release our code at `https://github.com/myt517/DKT`.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 46 - 53
May 22-27, 2022 ©2022 Association for Computational Linguistics

fied contrastive learning framework. Different from existing OOD discovery work, we equip the traditional IND pre-training stage with a similar contrastive objective as the clustering stage. Specifically, we firstly learn intent features using a context encoder like BERT, then add two independent transformation heads (instance-level head f and class-level head g) on top of BERT. In the IND pre-training stage, we use the head f to perform supervised instance-level contrastive learning (Chen et al., 2020; Khosla et al., 2020; Gunel et al., 2021; Zeng et al., 2021) and the head g to compute traditional classification loss like cross-entropy. In the OOD clustering stage, we employ similar objectives for these two heads where f is still used for instance-level contrastive learning and g is used to perform class(cluster)-level contrastive learning (Li et al., 2021). We leave the details in the following Section 2. Using the unified contrastive objectives for pre-training and clustering bridges the gap between the two stages. Besides, the two independent heads decouple the instance- and cluster-level contrastive learning to learn disentangled intent representations for better knowledge transfer. Section 4 demonstrates the effectiveness of multi-head disentanglement.

Our contributions are three-fold: (1) We propose a novel disentangled knowledge transfer method for OOD discovery to better leverage prior IND knowledge. (2) We propose a unified multi-head contrastive learning framework to bridge the gap between IND pre-training and OOD clustering. (3) Experiments and analysis on two benchmark datasets demonstrate the effectiveness of our method for OOD discovery.

2 Approach

Problem Formulation Given a set of labeled in-domain data $(\mathcal{X}_{IND}, \mathcal{Y}_{IND})$ and unlabeled OOD data $(\mathcal{X}_{OOD}, \mathcal{Y}_{OOD})$, OOD discovery aims to cluster OOD groups from unlabeled OOD data using prior knowledge from labeled IND data. Note that IND data has no overlapping with OOD data. Generally, OOD discovery includes two stages, IND pre-training which aims to obtain a decent intent representation via labeled IND data, and OOD clustering which aims to group OOD intents into different clusters.

Overall Architecture Fig 2 shows the overall architecture of our proposed DKT model. We firstly use the same BERT (Devlin et al., 2019) backbone to extract intent representations as the previous work DeepAligned (Zhang et al., 2021). Then we decouple the intent representations into two independent subspaces and use a unified contrastive learning framework to perform both IND pre-training and OOD clustering.

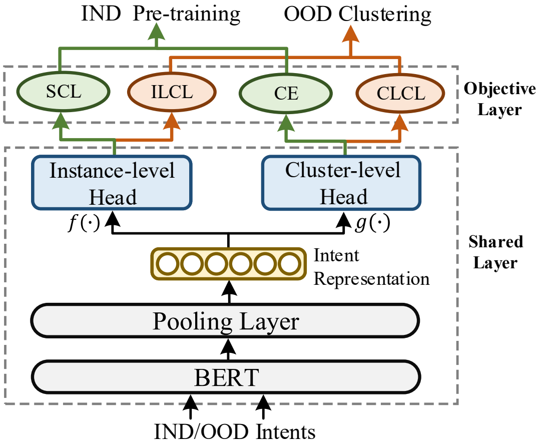


Figure 2: The overall architecture of our DKT.

IND Pre-training Different from existing methods that regard IND pre-training as a single intent classification task, we formulate it as an instance-wise discriminative task and a class-wise classification task via contrastive learning. Given an IND intent example x_i, we firstly obtain its intent representation z_i using a BERT encoder and a pooling layer.[2] Then we use two independent transformation heads f and g to get two disentangled latent vectors $f_i = f(z_i)$ and $g_i = g(z_i)$.[3] On top of the instance-level head f, we perform supervised contrastive learning (SCL) (Khosla et al., 2020; Zeng et al., 2021) as follows:

$$\mathcal{L}_{SCL} = \sum_{i=1}^{N} -\frac{1}{N_{y_i}-1} \sum_{j=1}^{N} \mathbf{1}_{i\neq j}\mathbf{1}_{y_i=y_j} \log \frac{\exp\left(f_i \cdot f_j/\tau\right)}{\sum_{k=1}^{N} \mathbf{1}_{i\neq k} \exp\left(f_i \cdot f_k/\tau\right)}$$

where N_{y_i} is the total number of examples in the batch that have the same label as y_i and $\mathbf{1}$ is an indicator function. Following Gao et al. (2021); Yan et al. (2021), we employ simple dropout (Srivastava et al., 2014) as data augmentation. SCL can model instance-wise semantic similarities by pulling together IND intents belonging to the same class while pushing apart samples from different

[2] For a fair comparison, we use the same BERT-based backbone as previous work. We leave the details to Section 3.4.

[3] In the experiments, we use two separate two-layer nonlinear MLPs for head f and g. For simplicity, we set both the input dimension and output dim to 768, same as the hidden state dim of BERT-base.

Models		CLINC-10%			CLINC-20%			CLINC-30%			Banking		
		ACC	ARI	NMI	ACC	ARI	NMI	ACC	ARI	NMI	ACC	ARI	NMI
Unsup.	K-means	58.67	43.81	67.77	48.89	30.90	64.68	42.22	23.65	60.55	32.81	8.30	17.30
	DeepCluster	53.15	37.80	62.31	47.73	34.55	65.91	33.96	18.89	56.21	29.81	7.79	17.34
	DeepAligned	62.66	47.60	71.50	48.24	34.49	66.24	39.02	24.50	61.16	36.56	12.57	21.84
	DKT(ours)	**74.22**	**61.37**	**76.67**	**57.56**	**44.94**	**72.40**	**50.07**	**35.53**	**69.81**	**40.00**	**18.20**	**30.10**
Semi-sup.	PTK-means	70.22	50.39	73.92	57.56	37.02	72.71	61.63	40.96	75.90	55.00	36.18	53.75
	DeepCluster	78.13	68.31	82.87	83.42	76.18	89.33	78.09	71.05	88.70	60.59	41.88	55.22
	CDAC+	88.00	75.18	88.33	84.89	75.98	89.96	73.04	64.44	87.90	77.50	60.53	71.14
	DeepAligned	95.11	89.81	94.13	93.80	90.22	95.83	91.56	86.58	94.91	77.78	66.95	76.91
	DKT(ours)	**97.78**	**95.16**	**96.97**	**96.89**	**93.69**	**96.94**	**94.96**	**90.25**	**95.94**	**84.69**	**71.11**	76.92

Table 1: Performance comparison on two datasets. We randomly sample 10%, 20% and 30% of all classes as OOD types for CLINC, 10% for Banking. We evaluate both unsupervised and semi-supervised methods. Unsup DKT denotes DKT w/o IND pre-training. Results are averaged over three random runs. ($p < 0.05$ under t-test)

classes. Therefore, SCL helps maximize inter-class variance and minimize intra-class variance, further improves OOD clustering. On top of the class-level head g, we use a cross-entropy classification loss to learn class(cluster)-wise distinction. Section 4 confirms both the objectives improve the performance and SCL has a larger effect.

OOD Clustering The key challenge of OOD clustering is how to learn intent representations and cluster assignments. Previous state-of-the-art model DeepAligned (Zhang et al., 2021) iteratively repeats the two stages which results in poor clustering efficiency and accuracy. Thus, we propose an end-to-end contrastive clustering method (Li et al., 2021) to jointly learn representations and cluster assignments. Specifically, given an OOD example x_i, we firstly use the pre-trained BERT encoder and transformation heads to get OOD intent latent vectors f_i and g_i. Then, on top of the instance-level head f, we perform instance-level contrastive learning(ILCL) (Chen et al., 2020) as follows:

$$\ell_{i,j}^{ins} = -\log \frac{\exp\left(\operatorname{sim}\left(f_i, f_j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbf{1}_{[k \neq i]} \exp\left(\operatorname{sim}\left(f_i, f_k\right)/\tau\right)}$$

where f_j denotes the dropout-augmented OOD sample and τ denotes temperature [4]. On top of the cluster-level head g, we perform contrastive clustering following Li et al. (2021) . Specifically, given an OOD cluster-level latent vector g_i, we firstly project it to a vector with dimension K which equals to the pre-defined cluster number.[5] Suppose we input a batch of OOD samples so we can get a feature matrix of $N \times K$. Then we regard i-th column of the matrix as the i-th cluster representation y_i and construct cluster-level CL(CLCL) as follows:

$$\ell_{i,j}^{clu} = -\log \frac{\exp\left(\operatorname{sim}\left(y_i, y_j\right)/\tau\right)}{\sum_{k=1}^{2K} 1_{[k \neq i]} \exp\left(\operatorname{sim}\left(y_i, y_k\right)/\tau\right)}$$

where y_j is the dropout-augmented cluster representation of y_i and sim denotes cosine distance. Following Li et al. (2021), we also add a regularization item to avoid the trivial solution that most instances are assigned to the single cluster. For training, we simply add the above objectives in the experiments. For inference, we only use the cluster-level contrastive head and compute the argmax to get the cluster results without additional K-means. Generally, the instance-CL focuses on distinguishing different intent samples while the cluster-CL identifies distinct OOD categories. Combining the two stages, our proposed unified contrastive learning framework can effectively bridge the gap between IND pre-training and OOD clustering.

3 Experiment

3.1 Datasets

We show the detailed statistics of CLINC(Larson et al., 2019) and BANKING(Casanueva et al., 2020) datasets in Table 2. CLINC contains 22,500 queries covering 150 intents and Banking contains 13,083 customer service queries with 77 intents. To construct IND/OOD data, we ramdomly divided the two datasets in three ramdom runs, according to the specified OOD ratio(10%, 20%, 30% for CLINC, 10% for Banking), and the rest is IND data. Note that we only use the IND data for pre-training and use OOD data for clustering. To avoid the randomness of splitting IND/OOD, we average results over three random runs. For each run, all the models use the same divided dataset. Different from previous work Zhang et al. (2021), we assume that the unlabeled data only contains OOD data instead of a mixture of IND and OOD, aiming to fairly evaluate the OOD clustering performance.

[4] we set it to 0.5 in the experiments.

[5] In this paper, we focus on the fixed cluster number K setting and leave estimating K to future work.

Dataset	Classes	Training	Validation	Test	Vocabulary	Length (max / mean)
CLINC	150	18,000	2,250	2,250	7,283	28 / 8.31
BANKING	77	9,003	1,000	3,080	5,028	79 / 11.91

Table 2: Statistics of CLINC and BANKING datasets.

In real scenarios, we can use OOD detection models (Xu et al., 2020; Zeng et al., 2021) to collect high-quality OOD data for OOD intent discovery.

3.2 Baselines

We mainly compare our method with semi-supervised baselines: PTK-means (k-means with IND pre-training), DeepCluster (Caron et al., 2018) and two state-of-the-art OOD discovery methods CDAC+ (Lin et al., 2020) and DeepAligned (Zhang et al., 2021). We also report the unsupervised results (without IND pretraining) of these methods for a comprehensive comparison. For fairness, we use the same BERT backbone as the baselines. We leave the detailed baselines in the appendix A.1.

3.3 Evaluation Metrics

We adopt three widely used metrics to evaluate the clustering results: Accuracy (ACC), Normalized Mutual Information (NMI), and Adjusted Rand Index (ARI). To calculate ACC, we use the Hungarian algorithm (Kuhn, 1955) to obtain the mapping between the predicted classes and ground-truth classes.

3.4 Implementation Details

For a fair comparison with previous work, we use the pre-trained BERT model (bert-base-uncased [6], with 12-layer transformer) as our network backbone, and add a pooling layer to get intent representation(dimension=768). Moreover, we freeze all but the last transformer layer parameters to achieve better performance with BERT backbone, and speed up the training procedure as suggested in (Zhang et al., 2021). During the pre-training phase, the training batch size is 128, and during the clustering phase, the training batch size is 512 for CLINC-10%, CLINC-30%, Banking-10%, and 400 for CLINC-20%. The learning rate is 5e-5 in the pre-training phase and 0.0003 in the clustering phase. Notably, We use dropout (Gao et al., 2021) to construct augmented examples for contrastive learning with dropout rate 0.1. For the instance-level contrastive head, the dimensionality of the row space is set to 128, and the temperatures of SCL and instance-level CL are 0.5. As for the cluster-level contrastive head, the dimensionality of the column space is naturally set to the number of IND classes/OOD clusters, and the cluster-level temperature parameter τ= 1.0 is used for all datasets. We use SC of validation OOD data (still unlabeled data) to choose the best checkpoint. The pre-training stage of our model lasts about 30 minutes and clustering runs for 10 minutes on CLINC-10%, both using a single Tesla T4 GPU(16 GB of memory).

3.5 Main Results

Table 1 shows the performance comparison of different models on two datasets. Under both unsupervised and semi-supervised settings, our proposed DKT consistently outperforms all the baselines. In this paper, we mainly focus on the latter setting. For the Semi-sup setting on CLINC-10%, DKT outperforms the previous state-of-the-art DeepAligned by 2.67%(ACC), 5.35%(ARI), 2.84%(NMI). Similar improvements are observed on other datasets. The results prove the effectiveness of our proposed disentangled knowledge transfer for OOD discovery. Comparing Unsup DKT with Semi-sup DKT, the latter significantly outperforms the former by 23.56%(ACC), 33.79%(ARI), 20.30%(NMI), which demonstrates the effectiveness of IND pre-training(see details in appendix A.2).

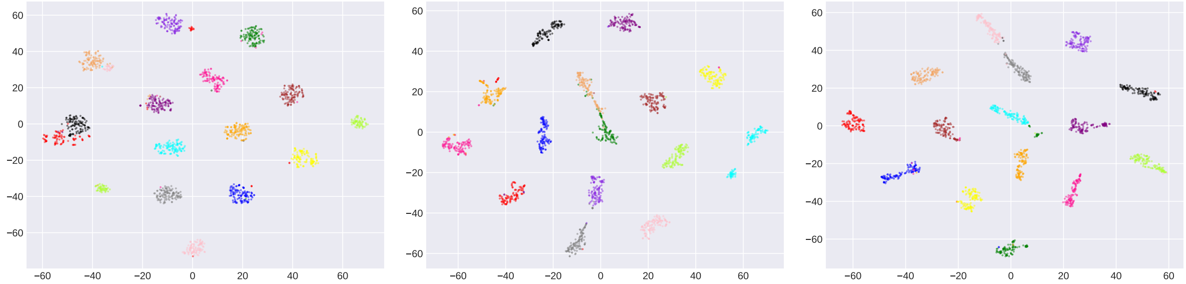
4 Qualitative Analysis

Effect of Disentangled Intent Representations Tab 3 shows performance comparison of DKT and KT under two settings. We find Disentangled KT significantly outperforms KT both on two settings, which proves the effectiveness of representation disentanglement for knowledge transfer.

Visualization To confirm the effectiveness of DKT, we perform OOD intent representation visualization of DeepAligned, KT and DKT in Fig 3. Note that we use the same representation following the pooling layer for fair comparison. We find both DeepAligned and KT have some mixed OOD clusters while DKT forms clearly separate decision boundaries between clusters, which shows our proposed DKT obtains discriminative OOD representations for OOD discovery. Besides, Section 4 further explore the effect of different layer and representations after MLP g gets the best performance.

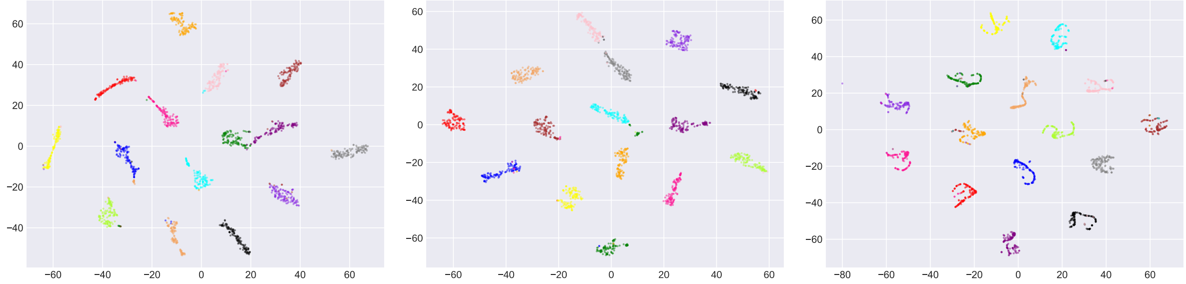
Error Analysis We further analyze the error cases of DeepAligned and DKT in Fig 5. We find that for

[6] https://github.com/google-research/bert



(a) DeepAligned (b) KT (c) DKT

Figure 3: Visualization of different methods. KT denotes only using single MLP head.



(a) Instance-level head (b) BERT + pooling layer (c) Cluster-level head

Figure 4: Intent representations at different layers

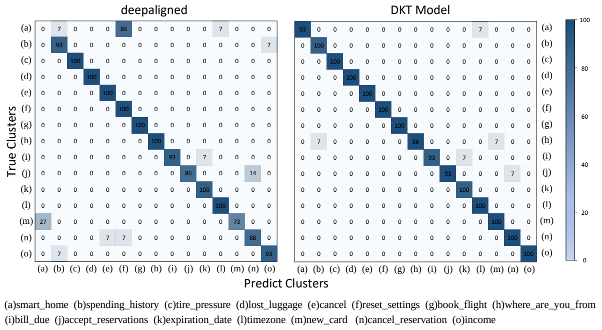


Figure 5: Confusion matrix for the clustering results of DeepAligned and DKT on CLINC-10%. The percentage values along the diagonal represent how many samples are correctly clustered into the corresponding class. The larger the number, the deeper the color.

similar OOD intents, DeepAligned is probably confused but our DKT can effectively distinguish them. For example, DeepAligned incorrectly groups *accept_reservation* intents into *cancel_reservation* (14% error rate) vs DKT(7%), which proves DKT helps separate semantically similar OOD intents.

Ablation Study To understand the effect of different objectives of DKT, we perform ablation study in Tab 4 by removing each loss. Results show all the losses contribute to the performance especially SCL, ILCL and CLCL, which confirms the effectiveness of our unified contrastive framework.

Intent Representations at Different Layers In order to further explore the effectiveness of disentangled representation, we visualize the output vectors of instance-level head and cluster-level head and compare them with the output vector after BERT + pooling in Fig 4. We can find that the output obtained by instance-level head forms a narrow and long cluster distribution, while the output obtained by cluster-level head forms a more compact and uniform cluster distribution. We argue that this reflects the effect of decoupling, that is, instance-level head decouples the uniqueness of each sample, and cluster-level head decouples the category characteristics of each sample.

Models		ACC	ARI	NMI
Unsup.	KT	68.89	56.33	73.93
	DKT	74.22	61.37	76.67
Semi-sup.	KT	95.11	90.23	94.53
	DKT	**97.78**	**95.16**	**96.97**

Table 3: Effect of disentangled intent representations.

Models	ACC	ARI	NMI
DKT	**97.78**	**95.16**	**96.97**
-w/o SCL	92.26	86.33	92.62
-w/o CE	95.16	90.61	94.80
-w/o ILCL	90.93	85.43	92.07
-w/o CLCL	90.36	82.91	90.55

Table 4: Effect of different learning objectives.

5 Conclusion

In this paper, we propose a novel disentangled knowledge transfer method (DKT) via a unified multi-head contrastive learning framework to transfer disentangled IND intent representations to OOD clustering. Experiments and analysis on two benchmarks demonstrate the effectiveness of DKT for OOD discovery. We hope to explore more self-supervised representation learning methods for OOD discovery in the future.

Acknowledgements

We thank all anonymous reviewers for their helpful comments and suggestions. This work was partially supported by National Key R&D Program of China No. 2019YFF0303300 and Subject II No. 2019YFF0303302, DOCOMO Beijing Communications Laboratories Co., Ltd, MoE-CMCC "Artifical Intelligence" Project No. MCM20190701.

Broader Impact

Task-oriented dialogue systems have demonstrated remarkable performance across a wide range of applications, with the promise of a significant positive impact on human production mode and lifeway. Intent classification is an important component of Task-oriented dialogue system. The existing intent classification models follow a closed set assumption and can only identify a limited number of predefined intent types. However, the real world is open. During the online deployment of dialogue system, out-of-domain (OOD) or unknown intents will appear continually. Recently, out-of-domain intent detection task has been widely studied, which can be used to collect these new intent data. The OOD intent discovery task studied in this paper is to make further use of these new intent data. It aims to cluster these OOD samples according to intents, so as to mine new intent types automatically, guide the future development of the system, and expand the classification ability of intent classification models.

References

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Jianlong Chang, L. Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. 2017. Deep adaptive image clustering. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5880–5888.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709.

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

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *ArXiv*, abs/2104.08821.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Ves Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. *ArXiv*, abs/2011.01403.

Dilek Hakkani-Tür, Yun-Cheng Ju, Geoffrey Zweig, and Gokhan Tur. 2015. Clustering novel intents in a conversational interaction system with semantic parsing. In *Sixteenth Annual Conference of the International Speech Communication Association*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *ArXiv*, abs/2004.11362.

H. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. 2021. Contrastive clustering. In *2021 AAAI Conference on Artificial Intelligence (AAAI)*.

Ting-En Lin, Hua Xu, and Hanlei Zhang. 2020. Discovering new intents via constrained deep adaptive clustering with cluster refinement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8360–8367.

J. MacQueen. 1967. Some methods for classification and analysis of multivariate observations.

Padmasundari and S. Bangalore. 2018. Intent discovery through unsupervised semantic text clustering. In *INTERSPEECH*.

Chen Shi, Qi Chen, Lei Sha, Sujian Li, Xu Sun, Houfeng Wang, and Lintao Zhang. 2018. Autodialabel: Labeling dialogue data with unsupervised learning. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 684–689.

Nitish Srivastava, Geoffrey E. Hinton, A. Krizhevsky, Ilya Sutskever, and R. Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958.

Junyuan Xie, Ross B. Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. *ArXiv*, abs/1511.06335.

Hong Xu, Keqing He, Yuanmeng Yan, Sihong Liu, Zijun Liu, and Weiran Xu. 2020. A deep generative distance-based classifier for out-of-domain detection with mahalanobis space. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1452–1460, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. In *ACL/IJCNLP*.

Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Zijun Liu, Yanan Wu, Hong Xu, Huixing Jiang, and Weiran Xu. 2021. Modeling discriminative representations for out-of-domain detection with supervised contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 870–878, Online. Association for Computational Linguistics.

Hanlei Zhang, Hua Xu, Ting-En Lin, and Rui Lyu. 2021. Discovering new intents with deep aligned clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14365–14373.

A Appendix

A.1 Baselines

The details of baselines are as follows:

- **PTK-means** A method based on k-means with IND pre-training. And the IND pre-training objectives uses CE + SCL proposed in this paper.

- **DeepCluster** An iterative clustering algorithm proposed by (Caron et al., 2018), in each iteration, firstly, k-means is used to assign pseudo label to the unlabeled samples, and then the cross-entropy objective is used for representation learning. The cluster header parameters need to be reinitialized during each iteration. In the semi-supervised setting, we use the same IND pre- training objective as DeepAligned (Zhang et al., 2021)

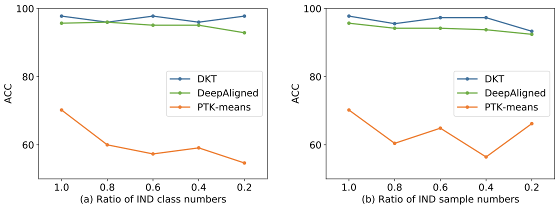


Figure 6: Effect of IND Data.

- **CDAC+** The first work of new intent discovery proposed by (Lin et al., 2020), and it firstly pre-trains a BERT-based (Devlin et al., 2019) in-domain intent classifier then uses intent representations to calculate the similarity of OOD intent pairs as weak supervised signals.

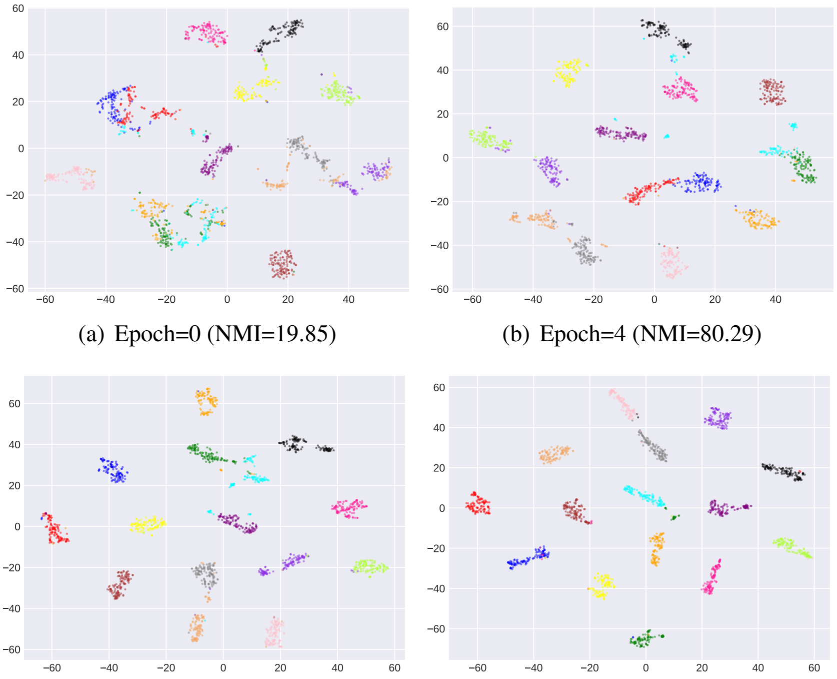
- **DeepAligned** The second work of new intent discovery proposed by (Zhang et al., 2021).It is an improved version of DeepCluster. It designed a pseudo label alignment strategy to produce aligned cluster assignments for better representation learning.

A.2 Effect of IND Data

We analyze the effect of IND data for OOD discovery from two perspectives, the number of IND classes and samples per class. Figure 6(a) shows the trend of the number of different IND classes, and Figure 6(b) shows the trend of the number of different samples in each class. Results show DKT outperforms baselines under all settings and gets the smallest varying degrees of performance drop, which proves the robustness and stability of our method.

A.3 Visualization at Different Training Epochs

To see the evolution of our method in the training process, we show a visualization at four different timestamps throughout the training process in Fig 7. Results show representation vector of different intent classes are mixed in the beginning and cluster assignments become increasingly visible and distinct as the training process goes.



(a) Epoch=0 (NMI=19.85)

(b) Epoch=4 (NMI=80.29)

(c) Epoch=8 (NMI=92.32)

(d) Epoch=14 (NMI=96.97)

Figure 7: OOD intent visualization of different training epochs for our proposed DKT method.

Voxel-informed Language Grounding

Rodolfo Corona **Shizhan Zhu** **Dan Klein** **Trevor Darrell**
Computer Science Division, University of California, Berkeley
{rcorona, shizhan_zhu, klein, trevordarrell}@berkeley.edu

Abstract

Natural language applied to natural 2D images describes a fundamentally 3D world. We present the Voxel-informed Language Grounder (VLG), a language grounding model that leverages *3D geometric information* in the form of voxel maps derived from the visual input using a volumetric reconstruction model. We show that VLG significantly improves grounding accuracy on SNARE (Thomason et al., 2021), an object reference game task. At the time of writing, VLG holds the top place on the SNARE leaderboard,[1] achieving SOTA results with a 2.0% absolute improvement.

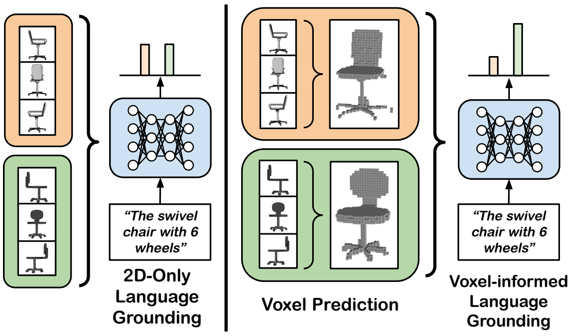

"The swivel chair with 6 wheels"
2D-Only Language Grounding
Voxel Prediction
"The swivel chair with 6 wheels"
Voxel-informed Language Grounding

Figure 1: **Voxel-informed Language Grounder.** Our VLG model leverages explicit 3D information by inferring volumetric voxel maps from input images, allowing the agent to reason jointly over the geometric and visual properties of objects when grounding.

1 Introduction

Embodied robotic agents hold great potential for providing assistive technologies in home environments (Pineau et al., 2003), and natural language provides an intuitive interface for users to interact with such systems (Andreas et al., 2020). For these systems to be effective, they must be able to reliably ground language in perception (Bisk et al., 2020; Bender and Koller, 2020).

Despite typically being paired with 2D images, natural language that is grounded in vision describes a fundamentally 3D world. For example, consider the grounding task in Figure 1, where the agent must select a target chair against a distractor given the description "the swivel chair with 6 wheels." Although the agent is provided with multiple images revealing all of the wheels on each chair, it must be able to properly aggregate information across images to successfully differentiate them, something that requires reasoning about their *3D geometry* at some level.

In this work, we show how language grounding performance may be improved by leveraging 3D prior knowledge. Our model, Voxel-informed Language Grounder (VLG), extracts 3D voxel maps using a pre-trained *volumetric reconstruction model*, which it fuses with multimodal features from a large-scale vision and language model in order to reason jointly over the visual and 3D geometric properties of objects.

We focus our investigation within the context of SNARE (Thomason et al., 2021), an object reference game where an agent must ground natural language describing common household objects by their geometric and visual properties, showing that grounding accuracy significantly improves by incorporating information from predicted 3D volumes of objects. At the time of writing, VLG achieves SOTA performance on SNARE, attaining an absolute improvement of 2.0% over the next closest baseline. Code to replicate our results is publicly available.[2]

2 Related Work

Prior work has studied deriving structured representations from images to scaffold language grounding. However, a majority of systems use representations such as 2D regions of interest (Anderson et al., 2018; Wang et al., 2020) or symbolic graph-

[1] https://github.com/snaredataset/snareleaderboard

[2] https://github.com/rcorona/voxel_informed_language_grounding

based representations (Hudson and Manning, 2019; Kulkarni et al., 2013), which do not encode 3D properties of objects.

Most prior work tying language to 3D representations has largely focused on generating 3D structures conditioned on language, rather than using them as intermediate representations for language grounding as we do here. Specifically, prior work has performed language conditioned generation at the scene (Chang et al., 2014, 2015a), pose (Ahuja and Morency, 2019; Lin et al., 2018), or object (Chen et al., 2018) level. More recently, a line of work has explored referring expression grounding in 3D by mapping referring expressions of objects to 3D bounding boxes localizing them in point clouds of indoor scenes (Achlioptas et al., 2020; Chen et al., 2020; Zhao et al., 2021; Roh et al., 2022). Standard approaches follow a two-tiered process where an object proposal system will first provide bounding boxes for candidate objects, and a scoring module will then compute a compatibility score between each box and the referring expression in order to ground it. At a more granular level, Koo et al. (2021) learn alignments from language to object parts by training agents on a reference game over point cloud representations of objects.

In contrast, in this work we focus on augmenting language grounding over 2D RGB images using structured 3D representations derived from them. For the task of visual language navigation, prior work has shown how a persistent 3D semantic map may be used as an intermediate representation to aid in selecting navigational waypoints (Chaplot et al., 2020; Blukis et al., 2021). The semantic maps, however, represent entire scenes with individual voxels representing object categories, rather than their geometry. In this work, we show how a more granular occupancy map representing objects' geometry can improve language grounding performance.

Closest to our work is that of Prabhudesai et al. (2020), which presents a method for mapping language to 3D features within scenes from the CLEVR (Johnson et al., 2017) dataset. Their system generates 3D feature maps inferred from images and then grounds language directly to 3D bounding boxes or coordinates. Their method assumes, however, that dependency parse trees are provided for the natural language inputs, and it is trained with supervised alignments between noun phrases and the 3D representations, which VLG does not require.

3 Voxel-informed Language Grounder

We consider a task where an agent must correctly predict a target object v^t against a distractor v^c given a natural language description $w^t = \{w_1, ..., w_m\}$ of the target. For each object, the agent is provided with n 2D views $v = \{x_1, ..., x_n\}$, $x_i \in \mathbb{R}^{3\times W\times H}$.

An agent for this task is represented by a scoring function $s(v, w) \in [0, 1]$, computing the compatibility between the target description and the 2D views of each object, and is used to select the maximally scoring candidate. We first use unimodal encoders to encode the language description into $e_w = h(w)$ and the object view images into a single aggregate visual embedding $e_v = g(v)$ before fusing them with a visiolinguistic module $e_{vw} = f_{vw}([e_v; e_w])$. Prior approaches to this problem (Thomason et al., 2021) directly input this fused representation to a scoring module to produce a score $s(e_{vw})$. They do not explicitly reason about the 3D properties of the observed objects, requiring the models to learn them implicitly.

In contrast, our Voxel-informed Language Grounder augments the scoring function s with explicit 3D volumetric information $e_o = o(v)$ extracted from a pre-trained multiview reconstruction model. The volumetric information (in the form of a factorization of a voxel occupancy map in $\mathbb{R}^{W\times H\times D}$) is first fused into a joint representation with the language using a multimodal voxel-language module $e_{ow} = f_{ow}([e_o; e_w])$. The scoring function then produces a score based on all three modalities $s([e_{vw}; e_{ow}])$.

3.1 Model Architecture

VLG (Figure 2) consists of two branches: a visiolinguistic module for fusing language and 2D RGB features, and a voxel-language module for fusing language with 3D volumetric features. A scoring function is then used to reason jointly over the output of the two branches, producing a compatibility score.

Visiolinguistic Module. The architecture of our visiolinguistic module f_{vw} (left panel, Figure 2) largely mirrors the architecture of MATCH from Thomason et al. (2021). A pre-trained CLIP-ViT (Radford et al., 2021) model is used to

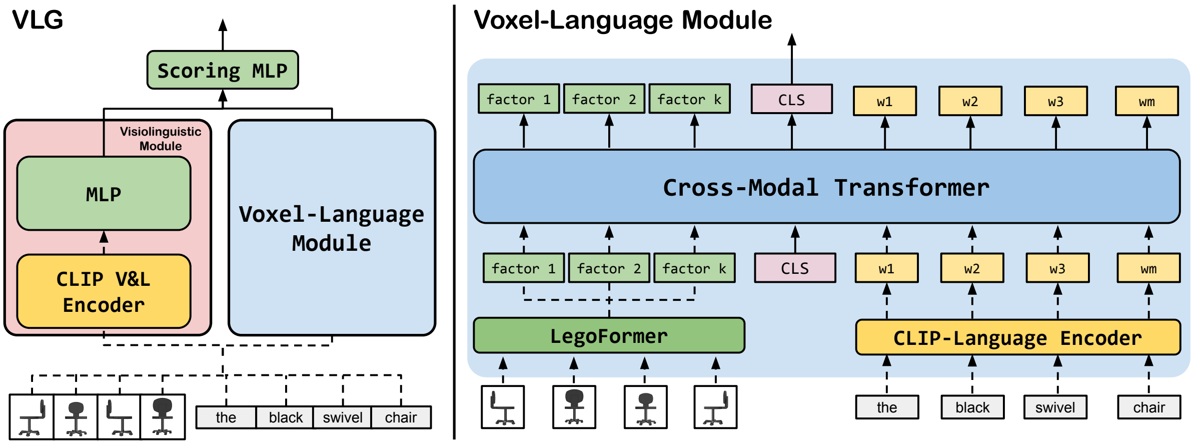


Figure 2: **VLG Architecture.** (Left) Our VLG model consists of a visiolinguistic module which produces a joint embedding for text and images using CLIP (Radford et al., 2021) and a voxel-language module for jointly embedding language and volumetric maps. (Right) The voxel-language module uses a cross modal transformer to fuse word embeddings from CLIP with voxel map factors extracted from LegoFormer (Yagubbayli et al., 2021). During training, gradients only flow through solid lines.

encode the language description and view images into vectors in $\mathbb{R}^{512}$. The image embeddings are max-pooled and concatenated to the description embedding before being passed into an MLP which generates a fused representation.

Voxel-Language Module. We use representations extracted from a ShapeNet (Chang et al., 2015b; Wu et al., 2015) pre-trained LegoFormerM (Yagubbayli et al., 2021), a multi-view 3D volumetric reconstruction model, as input to our voxel-language module f_{ow}. LegoFormer is a transformer (Vaswani et al., 2017) based model whose decoder generates volumetric maps factorized into 12 parts. Each object factor is represented by a set of three vectors $x, y, z \in \mathbb{R}^{32}$, which we concatenate to use as input tokens for our voxel-language module. A triple cross-product over x, y, z may be used to recover a 3D volume $\mathcal{V} \in \mathbb{R}^{32\times32\times32}$ for each factor. The full volume for the object is generated by aggregating the factor volumes through a sum operation. For more details on LegoFormer, we refer the reader to Yagubbayli et al. (2021). We use a cross-modal transformer (Vaswani et al., 2017) encoder to fuse the language and object factors (Figure 2, right). The cross-modal transformer takes as input language tokens, in the form of CLIP word embeddings, and the 12 object factors output by the LegoFormer decoder, which contain the inferred geometric occupancy information of the object. We use a CLS token as an aggregate representation of the language and object factors.

Scoring Function. The scoring function is represented by an MLP which takes as input the concatenation of the visiolinguistic module output and the cross-modal transformer's CLS token.

4 Language Grounding Evaluation

Evaluation. We test our method on the SNARE benchmark (Thomason et al., 2021). SNARE is a language grounding dataset which augments ACRONYM (Eppner et al., 2021), a grasping dataset built off of ShapeNetSem (Savva et al., 2015; Chang et al., 2015a), with natural language annotations of objects.

SNARE presents an object reference game where an agent must correctly guess a target object against a distractor. In each instance of the game, the agent is provided with a language description of the target as well as multiple 2D views of each object. SNARE differentiates between **visual** and **blindfolded** object descriptions. Visual descriptions primarily include attributes such as *name*, *shape*, and *color* (e.g. "classic armchair with white seat"). In contrast, blindfolded descriptions include attributes such as *shape* and *parts* (e.g. "oval back and vertical legs"). The train/validation/test sets were generated by splitting over (207 / 7 / 48) ShapeNetSem object categories, respectively containing (6,153 / 371 / 1,357) unique object instances and (39,104 / 2,304 / 8,751) object pairings with referring expressions. Renderings are provided for each object

	VALIDATION			TEST		
Model	Visual	Blind	All	Visual	Blind	All
ViLBERT	89.5	76.6	83.1	80.2	**73.0**	76.6
MATCH	89.2 (0.9)	75.2 (0.7)	82.2 (0.4)	83.9 (0.5)	68.7 (0.9)	76.5 (0.5)
MATCH*	90.6 (0.4)	75.7 (1.2)	83.2 (0.8)	-	-	-
LAGOR	89.8 (0.4)	75.3 (0.7)	82.6 (0.4)	84.3 (0.4)	69.4 (0.5)	77.0 (0.5)
LAGOR*	89.8 (0.5)	75.0 (0.4)	82.5 (0.1)	-	-	-
VLG (Ours)	**91.2** (0.4)	**78.4**†(0.7)	**84.9**†(0.3)	**86.0**	71.7	**79.0**

Table 1: **SNARE Benchmark Performance.** Object reference game accuracy on the SNARE task across validation and test sets. Performance on models with an asterisk are our replications of the baselines in Thomason et al. (2021). Standard deviations over 3 seeds are shown in parentheses. MATCH corresponds to the max-pool variant from Thomason et al. (2021) since no test set results are provided for the mean-pool variant. Our VLG model achieves the best overall performance. Due to leaderboard submission restrictions, we were not able to get test set results for the MATCH* and LAGOR* replications. † denotes statistical significance with $p < 0.1$.

instance over 8 canonical viewing angles.

Because ShapeNet and ShapeNetSem represent different splits of the broader ShapeNet database, we pre-train the LegoFormerM model on a modified dataset to avoid dataset leakage. Specifically, any objects which appear in both datasets are reassigned within the pre-training dataset used to train LegoFormerM to match its split assignment from SNARE.

ShapeNetSem images are resized to 224 × 224 when inputting them to LegoFormerM in order to match its ShapeNet pre-training conditions.

Baselines. We compare VLG against the set of models provided with SNARE.[3] All SNARE baselines except **ViLBERT** use a CLIP ViT-B/32 (Radford et al., 2021) backbone for encoding both images and language descriptions:

> **MATCH** first uses CLIP-ViT to embed the language description as well as each of the 8 view images. Next, the view embeddings are mean-pooled and concatenated to the description embedding. Finally, a learned MLP is used over the concatenated feature vector in order to produce a final compatibility score.
>
> **ViLBERT** fine-tunes a 12-in-1 (Lu et al., 2020) pre-trained ViLBERT(Lu et al., 2019) as the backbone for MATCH instead of using CLIP-ViT. Each object is presented to ViLBERT in the form of a single tiled image containing all 14 views from ShapeNetSem, instead of just the canonical 8 presented in the standard task. ViLBERT tokenizes images by extracting features from image regions, with the ground truth bounding boxes for each region (i.e. view) being provided. Because this baseline is not open-source, we report the original numbers from Thomason et al. (2021).
>
> **LAGOR** (**La**nguage **G**rounding through **O**bject **R**otation) fine-tunes a pre-trained MATCH module and is additionally regularized through the auxiliary task of predicting the canonical viewing angle of individual view images, which it predicts using an added output MLP head. Following Thomason et al. (2021), the LAGOR baseline is only provided with 2 random views of each object both during training and inference.

For more details on the baseline models, we refer the reader to Thomason et al. (2021).

Training Details. Apart from the dataset split re-assignments mentioned in Section 4, we use the code[4] and hyperparameters presented by Yagubbayli et al. (2021) to train LegoFormerM.

For training on SNARE, we follow Thomason et al. (2021) and train all models with a smoothed binary cross-entropy loss (Achlioptas et al., 2019).

We train each model for 75 epochs, reporting performance of the best performing checkpoint on the validation set. For our replication of the SNARE MATCH and LAGOR baselines, we use the code and hyperparameters provided by Thomason et al. (2021). For all variants of our VLG model we use the AdamW (Loshchilov and Hutter, 2017) optimizer with a learning rate of 1e-3 and a linear learning rate warmup of 10K steps.

[3] https://github.com/snaredataset/snare

[4] https://github.com/faridyagubbayli/LegoFormer

Model	Visual	Blind	All
VGG16	**91.4** (0.5)	76.5 (0.9)	84.0 (0.2)
MLP	91.1 (0.8)	77.9 (0.9)	84.6 (0.1)
no-CLIP	71.0 (0.6)	65.8 (0.7)	68.4 (0.1)
VLG	91.2 (0.4)	**78.4** (0.7)	**84.9** (0.3)

Table 2: **Ablation Study.** SNARE reference game accuracy across ablations of our model on the validation set. We show performance when replacing LegoformerM object factors with **VGG16** features, replacing the cross-modal transformer with an **MLP**, and when foregoing the use of CLIP features (**no-CLIP**).

5 Results

We present test set performance for VLG and the SNARE baselines reported by Thomason et al. (2021). We also present average performance for trained models over 3 seeds with standard deviations on the validation set.

5.1 Comparison to SOTA

In Table 1 we can observe reference game performance for all models. VLG achieves SOTA performance with an absolute improvement on the test set of 2.0% over LAGOR, the next best leaderboard model. Although there is a general improvement of 1.7% in **visual** reference grounding, there is an improvement of 2.3% in **blindfolded** (denoted as **Blind** in tables to conserve space) reference grounding. This suggests that the injected 3D information provides a greater boost for disambiguating between examples referring to geometric properties of target objects. VLG generally improves over all baselines and conditions for blindfolded examples, with the exception of ViLBERT, which may be due to the additional information ViLBERT receives in the form of 14 viewing angles of each object instead of 8.

Improvements on the Blind and All conditions of the validation set are statistically significant with $p < 0.1$ under a Welch's two-tailed t-test.

5.2 Ablation Study

We present a variety of ablations on the validation set to investigate the contributions of each piece of our model. All results can be observed in Table 2.

VGG16 Embeddings. LegoFormer uses an ImageNet (Deng et al., 2009) pre-trained VGG16 (Simonyan and Zisserman, 2014) as a backbone for extracting visual representations, which is a different dataset and pre-training task than what the CLIP-ViT image encoder is trained on. This presents a confounding factor which we ablate by performing an experiment feeding our model's scoring function VGG16 features directly instead of LegoFormer object factors (VGG16 in Table 2). Despite getting comparable results to VGG16 on visual reference grounding, VLG provides a clear improvement in blindfolded (and therefore overall) reference performance, suggesting that the extracted 3D information is useful for grounding more geometrically based language descriptions, with the VGG16 features being largely redundant in terms of visual signal.

Architecture. We ablate the contribution of our cross-modal transformer branch by comparing it against an MLP mirroring the structure of the SNARE MATCH baseline. This model (MLP in Table 2) max-pools the LegoFormer object factors and concatenates the result to the CLIP visual and language features before passing them to an MLP scoring function. The MLP model overall outperforms the SNARE baselines from Table 1, highlighting the usefulness of the 3D information for grounding, but does not result in as large an improvement as the cross-modal transformer. This suggests that the transformer is better able at integrating information from the multi-view input.

CLIP Visual Embeddings. Finally, we evaluate the contribution of the visiolinguistic branch of the model by removing it and only using the cross-modal transformer over language and object factors. As may be observed, there is a large drop in performance (16.5% overall), particularly for visual references (20.2%). These results suggest that maintaining visual information such as color and texture is critical for performing well on this task, since the LegoFormer outputs contain only volumetric occupancy information.

6 Discussion

We have presented the Voxel-informed Language Grounder (VLG), a model which leverages explicit 3D information from predicted volumetric voxel maps to improve language grounding performance. VLG achieves SOTA results on SNARE, and ablations demonstrate the effectiveness of using this 3D information for grounding. We hope this paper may inspire future work on integrating structured 3D representations into language grounding tasks.

Acknowledgements

We would like to thank Karttikeya Mangalam and Nikita Kitaev for their helpful advice and discussions on transformer models. Mohit Shridhar and Jesse Thomason for their help with setting up SNARE. And thanks to the anonymous reviewers for their constructive feedback. This work was supported by DARPA under the SemaFor program (HR00112020054). The content does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred. RC is supported by an NSF Graduate Research Fellowship.

References

Panos Achlioptas, Ahmed Abdelreheem, Fei Xia, Mohamed Elhoseiny, and Leonidas J. Guibas. 2020. ReferIt3D: Neural listeners for fine-grained 3d object identification in real-world scenes. In *16th European Conference on Computer Vision (ECCV)*.

Panos Achlioptas, Judy Fan, Robert Hawkins, Noah Goodman, and Leonidas J Guibas. 2019. Shapeglot: Learning language for shape differentiation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8938–8947.

Chaitanya Ahuja and Louis-Philippe Morency. 2019. Language2pose: Natural language grounded pose forecasting. In *2019 International Conference on 3D Vision (3DV)*, pages 719–728. IEEE.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.

Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, et al. 2020. Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

Emily M Bender and Alexander Koller. 2020. Climbing towards nlu: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, et al. 2020. Experience grounds language. *arXiv preprint arXiv:2004.10151*.

Valts Blukis, Chris Paxton, Dieter Fox, Animesh Garg, and Yoav Artzi. 2021. A persistent spatial semantic representation for high-level natural language instruction execution. *arXiv preprint arXiv:2107.05612*.

Angel Chang, Manolis Savva, and Christopher D Manning. 2014. Learning spatial knowledge for text to 3d scene generation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2028–2038.

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015a. Shapenet: An information-rich 3d model repository. Cite arxiv:1512.03012.

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015b. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago.

Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. 2020. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33.

Dave Zhenyu Chen, Angel X Chang, and Matthias Nießner. 2020. Scanrefer: 3d object localization in rgb-d scans using natural language. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 202–221. Springer.

Kevin Chen, Christopher B Choy, Manolis Savva, Angel X Chang, Thomas Funkhouser, and Silvio Savarese. 2018. Text2shape: Generating shapes from natural language by learning joint embeddings. In *Asian Conference on Computer Vision*, pages 100–116. Springer.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Clemens Eppner, Arsalan Mousavian, and Dieter Fox. 2021. Acronym: A large-scale grasp dataset based on simulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6222–6227. IEEE.

Drew A Hudson and Christopher D Manning. 2019. Learning by abstraction: The neural state machine. *arXiv preprint arXiv:1907.03950*.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference* *on computer vision and pattern recognition*, pages 2901–2910.

Juil Koo, Ian Huang, Panos Achlioptas, Leonidas Guibas, and Minhyuk Sung. 2021. Partglot: Learning shape part segmentation from language reference games. *arXiv preprint arXiv:2112.06390*.

Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. 2013. Babytalk: Understanding and generating simple image descriptions. *IEEE transactions on pattern analysis and machine intelligence*, 35(12):2891–2903.

Angela S Lin, Lemeng Wu, Rodolfo Corona, Kevin Tai, Qixing Huang, and Raymond J Mooney. 2018. Generating animated videos of human activities from natural language descriptions. *Learning*, 2018:1.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*.

Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 2020. 12-in-1: Multi-task vision and language representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10437–10446.

Joelle Pineau, Michael Montemerlo, Martha Pollack, Nicholas Roy, and Sebastian Thrun. 2003. Towards robotic assistants in nursing homes: Challenges and results. *Robotics and autonomous systems*, 42(3-4):271–281.

Mihir Prabhudesai, Hsiao-Yu Fish Tung, Syed Ashar Javed, Maximilian Sieb, Adam W Harley, and Katerina Fragkiadaki. 2020. Embodied language grounding with 3d visual feature representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2220–2229.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

Junha Roh, Karthik Desingh, Ali Farhadi, and Dieter Fox. 2022. Languagerefer: Spatial-language model for 3d visual grounding. In *Conference on Robot Learning*, pages 1046–1056. PMLR.

Manolis Savva, Angel X. Chang, and Pat Hanrahan. 2015. Semantically-Enriched 3D Models for Common-sense Knowledge. *CVPR 2015 Workshop on Functionality, Physics, Intentionality and Causality*.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Jesse Thomason, Mohit Shridhar, Yonatan Bisk, Chris Paxton, and Luke Zettlemoyer. 2021. Language grounding with 3d objects. In *5th Annual Conference on Robot Learning*.

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

Ruocheng Wang, Jiayuan Mao, Samuel J Gershman, and Jiajun Wu. 2020. Language-mediated, object-centric representation learning. *arXiv preprint arXiv:2012.15814*.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920.

Farid Yagubbayli, Alessio Tonioni, and Federico Tombari. 2021. Legoformer: Transformers for block-by-block multi-view 3d reconstruction. *arXiv preprint arXiv:2106.12102*.

Lichen Zhao, Daigang Cai, Lu Sheng, and Dong Xu. 2021. 3dvg-transformer: Relation modeling for visual grounding on point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2928–2937.

P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks

Xiao Liu[1,2*], Kaixuan Ji[1*], Yicheng Fu[1*], Weng Lam Tam[1], Zhengxiao Du[1,2], Zhilin Yang[1,3†], Jie Tang[1,2†]

[1]Tsinghua University, KEG [2]Beijing Academy of Artificial Intelligence (BAAI)
[3]Shanghai Qi Zhi Institute
`{liuxiao21,jkx19,fyc19}@mails.tsinghua.edu.cn`

Abstract

Prompt tuning, which only tunes continuous prompts with a frozen language model, substantially reduces per-task storage and memory usage at training. However, in the context of NLU, prior work reveals that prompt tuning does not perform well for normal-sized pretrained models. We also find that existing methods of prompt tuning cannot handle hard sequence labeling tasks, indicating a lack of universality. We present a novel empirical finding that properly optimized prompt tuning can be universally effective across a wide range of model scales and NLU tasks. It matches the performance of finetuning while having only 0.1%-3% tuned parameters. Our method P-Tuning v2 is an implementation of Deep Prompt Tuning (Li and Liang, 2021; Qin and Eisner, 2021) optimized and adapted for NLU. Given the universality and simplicity of P-Tuning v2, we believe it can serve as an alternative to finetuning and a strong baseline for future research.[1]

1 Introduction

Pretrained language models (Radford et al., 2019; Devlin et al., 2018; Yang et al., 2019; Raffel et al., 2019) improve performance on a wide range of natural language understanding (NLU) tasks. A widely-used method, **fine-tuning**, updates the entire set of model parameters for a target task. While fine-tuning obtains good performance, it is memory-consuming during training because gradients and optimizer states for all parameters must be stored. Moreover, keeping a copy of model parameters for each task during inference is inconvenient since pre-trained models are usually large.

Prompting, on the other hand, freezes all parameters of a pre-trained model and uses a natural language prompt to query a language model (Brown et al., 2020). For example, for sentiment analysis, we can concatenate a sample (e.g., "Amazing movie!") with a prompt “This movie is [MASK]” and ask the pre-trained language model to predict the probabilities of masked token being “good” and “bad” to decide the sample’s label. Prompting requires no training at all and stores one single copy of model parameters. However, discrete prompting (Shin et al., 2020; Gao et al., 2020) can lead to suboptimal performance in many cases compared to fine-tuning.

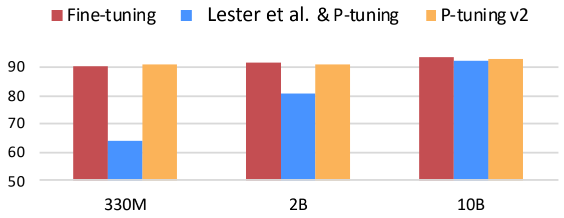


Figure 1: Average scores on RTE, BoolQ and CB of SuperGLUE dev. With 0.1% task-specific parameters, P-tuning v2 can match fine-tuning across wide scales of pre-trained models, while Lester et al. (2021) & P-tuning can make it conditionally at 10B scale.

Prompt tuning[2] is an idea of tuning only the continuous prompts. Specifically, Liu et al. (2021); Lester et al. (2021) proposed to add trainable continuous embeddings (also called continuous prompts) to the original sequence of input word embeddings. Only the continuous prompts are updated during training. While prompt tuning improves over prompting on many tasks (Liu et al., 2021; Lester et al., 2021; Zhong et al., 2021), it still underperforms fine-tuning when the model size is not large, specifically less than 10 billion parameters (Lester et al., 2021). Moreover, as shown in our experiments, prompt tuning performs poorly compared to fine-tuning on several hard sequence labeling tasks such as extractive question answering (Cf. Section 4.2).

[†] corresponding to: Zhilin Yang (zhiliny@tsinghua.edu.cn) and Jie Tang (jietang@tsinghua.edu.cn)

[*] indicates equal contribution.

[1] Our code and data are released at `https://github.com/THUDM/P-tuning-v2`.

[2] We use “prompt tuning” to refer to a class of methods rather than a particular method.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 61 - 68
May 22-27, 2022 ©2022 Association for Computational Linguistics

Our main contribution in this paper is a novel empirical finding that properly optimized prompt tuning can be comparable to fine-tuning universally across various model scales and NLU tasks. In contrast to observations in prior work, our discovery reveals the universality and potential of prompt tuning for NLU.

Technically, our approach P-tuning v2 is not conceptually novel. It can be viewed as an optimized and adapted implementation of **Deep Prompt Tuning** (Li and Liang, 2021; Qin and Eisner, 2021) designed for generation and knowledge probing. The most significant improvement originates from appling continuous prompts for every layer of the pretrained model, instead of the mere input layer. Deep prompt tuning increases the capacity of continuous prompts and closes the gap to fine-tuning across various settings, especially for small models and hard tasks. Moreover, we present a series of critical details of optimization and implementation to ensure finetuning-comparable performance.

Experimental results show that P-tuning v2 matches the performance of fine-tuning at different model scales ranging from 300M to 10B parameters and on various hard sequence tagging tasks such as extractive question answering and named entity recognition. P-tuning v2 has 0.1% to 3% trainable parameters per task compared to fine-tuning, which substantially reduces training time memory cost and per-task storage cost.

2 Preliminaries

NLU Tasks. In this work, we categorize NLU challenges into two families: *simple classification tasks* and *hard sequence labeling tasks*.[3] Simple classification tasks involve classification over a label space. Most datasets from GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019) are in this category. Hard sequence labeling tasks involve classification over a sequence of tokens, such as named entity recognition and extractive question answering.

Prompt Tuning. Let $\mathcal{V}$ be the vocabulary of a language model $\mathcal{M}$ and let $\mathbf{e}$ be the embedding layer of $\mathcal{M}$. In the case of discrete prompting (Schick and Schütze, 2020), prompt tokens $\{$"It", "is", "[MASK]"$\} \subset \mathcal{V}$ can be used to classify a movie review. For example, given the input text $\mathbf{x}$ ="Amazing movie!", the input embedding sequence is formulated as $[\mathbf{e}(\mathbf{x}), \mathbf{e}(\text{"It"}), \mathbf{e}(\text{"is"}), \mathbf{e}(\text{"[MASK]"})]$.

Lester et al. (2021) and Liu et al. (2021) introduce trainable continuous prompts as a substitution to natural language prompts for NLU with the parameters of pretrained language models frozen. Given the trainable continuous embeddings $[h_0, ..., h_i]$, the input embedding sequence is written as $[\mathbf{e}(\mathbf{x}), h_0, ..., h_i, \mathbf{e}(\text{"[MASK]"})]$, as illustrated in Figure 2. Prompt tuning has been proved to be comparable to fine-tuning on 10-billion-parameter models on simple classification tasks (Lester et al., 2021; Kim et al., 2021; Liu et al., 2021).

3 P-Tuning v2

3.1 Lack of Universality

Lester et al. (2021); Liu et al. (2021) have been proved quite effective in many NLP applications (Wang et al., 2021a,b; Chen et al., 2021; Zheng et al., 2021; Min et al., 2021), but still fall short at replacing fine-tuning due to lack of universality, as discussed below.

Lack of universality across scales. Lester et al. (2021) shows that prompt tuning can be comparable to fine-tuning when the model scales to over 10 billion parameters. However, for medium-sized models (from 100M to 1B) that are widely used, prompt tuning performs much worse than fine-tuning.

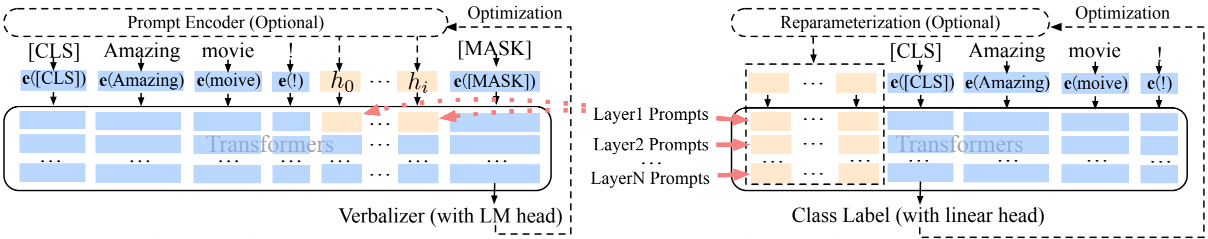
Lack of universality across tasks. Though Lester et al. (2021); Liu et al. (2021) have shown superiority on some of the NLU benchmarks, the effectiveness of prompt tuning on hard sequence tagging tasks is not verified. Sequence tagging predicts a sequence of labels for each input token, which can be harder and incompatible with verbalizers (Schick and Schütze, 2020). In our experiments (Cf. Section 4.2 and Table 3), we show that Lester et al. (2021); Liu et al. (2021) perform poorly on typical sequence tagging tasks compared to fine-tuning.

Considering these challenges, we propose P-tuning v2, which adapts deep prompt tuning (Li and Liang, 2021; Qin and Eisner, 2021) as a universal solution across scales and NLU tasks.

3.2 Deep Prompt Tuning

In (Lester et al., 2021) and (Liu et al., 2021), continuous prompts are only inserted into the input embedding sequence (Cf. Figure 2 (a)). This leads

[3] Note that the notions of “simple” and “hard” are specific to prompt tuning, because we find sequence labeling tasks are more challenging for prompt tuning.



(a) Lester et al. & P-tuning (Frozen, 10-billion-scale, simple tasks)

(b) P-tuning v2 (Frozen, most scales, most tasks)

Figure 2: From Lester et al. (2021) & P-tuning to P-tuning v2. Orange blocks (i.e., $h_0, ..., h_i$) refer to trainable prompt embeddings; blue blocks are embeddings stored or computed by frozen pre-trained language models.

to two challenges. First, the number of tunable parameters is limited due to the constraints of sequence length. Second, the input embeddings have relatively indirect impact on model predictions.

To address these challenges, P-tuning v2 employs the idea of deep prompt tuning (Li and Liang, 2021; Qin and Eisner, 2021). As illustrated in Figure 2, prompts in different layers are added as prefix tokens. On one hand, P-tuning v2 have more tunable task-specific parameters (from 0.01% to 0.1%-3%) to allow more per-task capacity while being parameter-efficient; on the other hand, prompts added to deeper layers have more direct impact on model predictions (see analysis in Appendix B).

3.3 Optimization and Implementation

There are a few useful details of optimization and implementation for achieving the best performance.

Reparameterization. Prior works usually leverage a reparameterization encoder such as an MLP (Li and Liang, 2021; Liu et al., 2021) to transform trainable embeddings. However, for NLU, we discover that its usefulness depends on tasks and datasets. For some datasets (e.g., RTE and CoNLL04), MLP brings a consistent improvement; for the others, MLP leads to minimal or even negative effects on the results (e.g., BoolQ and CoNLL12). See Appendix B for more analysis.

Prompt Length. The prompt length plays a critical role in P-Tuning v2. We find that different NLU tasks usually achieve their best performance with different prompt lengths (Cf. Appendix B). Generally, simple classification tasks prefer shorter prompts (less than 20); hard sequence labeling tasks prefer longer ones (around 100).

Multi-task Learning. Multi-task learning jointly optimizes multiple tasks with shared continuous prompts before fine-tuning for individual tasks. Multi-task is optional for P-Tuning v2 but can be used for further boost performance by providing a better initialization (Gu et al., 2021).

Method	Task	Re-param.	Deep PT	Multi-task	No verb.
P-tuning (Liu et al., 2021)	KP NLU	LSTM	-	-	-
PROMPTTUNING (Lester et al., 2021)	NLU	-	-	✓	-
Prefix Tuning (Li and Liang, 2021)	NLG	MLP	✓	-	-
SOFT PROMPTS (Qin and Eisner, 2021)	KP	-	✓	-	-
P-tuning v2 (Ours)	NLU SeqTag	*(depends)*	✓	✓	✓

Table 1: Conceptual comparison between P-tuning v2 and existing Prompt Tuning approaches (KP: Knowledge Probe; SeqTag: Sequence Tagging; Re-param.: Reparameterization; No verb.: No verbalizer).

Classification Head. Using a language modeling head to predict verbalizers (Schick and Schütze, 2020) has been central for prompt tuning (Liu et al., 2021), but we find it unnecessary in a full-data setting and incompatible with sequence labeling. P-tuning v2 instead applies a randomly-initialized classification head on top of the tokens as in BERT (Devlin et al., 2018) (Cf. Figure 2).

To clarify P-tuning v2's major contribution, we present a conceptual comparison to existing prompt tuning approaches in Table 1.

4 Experiments

We conduct extensive experiments over different commonly-used pre-trained models and NLU tasks to verify the effectiveness of P-tuning v2. In this work, all methods except for fine-tuning are conducted with **frozen language model backbones**, which accords with (Lester et al., 2021)'s setting but differs from (Liu et al., 2021)'s tuned setting. Ratios of task-specific parameters (e.g., 0.1%) are

	#Size	BoolQ			CB			COPA			MultiRC (F1a)		
		FT	PT	PT-2	FT	PT	PT-2	FT	PT	PT-2	FT	PT	PT-2
BERT$_{\text{large}}$	335M	**77.7**	67.2	75.8	**94.6**	80.4	**94.6**	69.0	55.0	**73.0**	70.5	59.6	**70.6**
RoBERTa$_{\text{large}}$	355M	**86.9**	62.3	84.8	98.2	71.4	**100**	**94.0**	63.0	93.0	**85.7**	59.9	82.5
GLM$_{\text{xlarge}}$	2B	**88.3**	79.7	87.0	**96.4**	76.4	**96.4**	**93.0**	92.0	91.0	84.1	77.5	**84.4**
GLM$_{\text{xxlarge}}$	10B	88.7	**88.8**	**88.8**	**98.7**	98.2	96.4	**98.0**	**98.0**	**98.0**	**88.1**	86.1	**88.1**

	#Size	ReCoRD (F1)			RTE			WiC			WSC		
		FT	PT	PT-2	FT	PT	PT-2	FT	PT	PT-2	FT	PT	PT-2
BERT$_{\text{large}}$	335M	70.6	44.2	**72.8**	70.4	53.5	**78.3**	74.9	63.0	**75.1**	**68.3**	64.4	**68.3**
RoBERTa$_{\text{large}}$	355M	89.0	46.3	**89.3**	86.6	58.8	**89.5**	**75.6**	56.9	73.4	63.5	**64.4**	63.5
GLM$_{\text{xlarge}}$	2B	91.8	82.7	**91.9**	**90.3**	85.6	**90.3**	**74.1**	71.0	72.0	**95.2**	87.5	92.3
GLM$_{\text{xxlarge}}$	10B	**94.4**	87.8	92.5	**93.1**	89.9	**93.1**	**75.7**	71.8	74.0	**95.2**	94.2	93.3

Table 2: Results on SuperGLUE development set. P-tuning v2 significantly surpasses P-tuning & Lester et al. (2021) on models smaller than 10B, and matches the performance of fine-tuning across different model scales. (FT: fine-tuning; PT: Lester et al. (2021) & P-tuning; PT-2: P-tuning v2; **bold**: the best; underline: the second best).

	#Size	CoNLL03				OntoNotes 5.0				CoNLL04			
		FT	PT	PT-2	MPT-2	FT	PT	PT-2	MPT-2	FT	PT	PT-2	MPT-2
BERT$_{\text{large}}$	335M	**92.8**	81.9	90.2	91.0	**89.2**	74.6	86.4	86.3	85.6	73.6	84.5	**86.6**
RoBERTa$_{\text{large}}$	355M	92.6	86.1	**92.8**	**92.8**	**89.8**	80.8	**89.8**	**89.8**	88.8	76.2	88.4	**90.6**
DeBERTa$_{\text{xlarge}}$	750M	**93.1**	90.2	**93.1**	**93.1**	90.4	85.1	90.4	**90.5**	89.1	82.4	86.5	**90.1**

	#Size	SQuAD 1.1 dev (EM / F1)								SQuAD 2.0 dev (EM / F1)							
		FT		PT		PT-2		MPT-2		FT		PT		PT-2		MPT-2	
BERT$_{\text{large}}$	335M	**84.2**	**91.1**	1.0	8.5	77.8	86.0	82.3	89.6	**78.7**	**81.9**	50.2	50.2	69.7	73.5	72.7	75.9
RoBERTa$_{\text{large}}$	355M	**88.9**	**94.6**	1.2	12.0	88.5	94.4	88.0	94.1	**86.5**	**89.4**	50.2	50.2	82.1	85.5	83.4	86.7
DeBERTa$_{\text{xlarge}}$	750M	90.1	95.5	2.4	19.0	**90.4**	**95.7**	89.6	95.4	88.3	91.1	50.2	50.2	**88.4**	**91.1**	88.1	90.8

	#Size	CoNLL12				CoNLL05 WSJ				CoNLL05 Brown			
		FT	PT	PT-2	MPT-2	FT	PT	PT-2	MPT-2	FT	PT	PT-2	MPT-2
BERT$_{\text{large}}$	335M	84.9	64.5.	83.2	**85.1**	**88.5**	76.0	86.3	**88.5**	82.7	70.0	80.7	**83.1**
RoBERTa$_{\text{large}}$	355M	**86.5**	67.2	84.6	86.2	**90.2**	76.8	89.2	90.0	85.6	70.7	84.3	**85.7**
DeBERTa$_{\text{xlarge}}$	750M	86.5	74.1	85.7	**87.1**	**91.2**	82.3	90.6	**91.2**	86.9	77.7	86.3	**87.0**

Table 3: Results on Named Entity Recognition (NER), Question Answering (Extractive QA), and Semantic Role Labeling (SRL). All metrics in NER and SRL are micro-f1 score. (FT: fine-tuning; PT: P-tuning & Lester et al. (2021); PT-2: P-tuning v2; MPT-2: Multi-task P-tuning v2; **bold**: the best; underline: the second best).

derived from comparing continuous prompts' parameters with transformers' parameters. Another thing to notice is that our experiments are all conducted in the fully-supervised setting rather than few-shot setting.

NLU Tasks. First, we include datasets from SuperGLUE (Wang et al., 2019) to test P-tuning v2's general NLU ability. Additionally, we introduce a suite of sequence labeling tasks, including named entity recognition (Sang and De Meulder, 2003; Weischedel et al., 2013; Carreras and Màrquez, 2004), extractive Question Answering (Rajpurkar et al., 2016), and semantic role labeling (Carreras and Màrquez, 2005; Pradhan et al., 2012)).

Pre-trained Models. We include BERT-large (Devlin et al., 2018), RoBERTa-large (Liu et al., 2019), DeBERTa-xlarge (He et al., 2020), GLM-xlarge/xxlarge (Du et al., 2021) for evaluation. They are all bidirectional models designed for NLU tasks, covering a wide range of sizes from about 300M to 10B.

Multitask Learning. For the multi-task setting, we combine the training sets of the datasets in each task type (e.g., combing all training sets of semantic role labeling). We use separate linear classifiers for each dataset while sharing the continuous prompts (Cf. Appendix A).

	SST-2	RTE	BoolQ	CB
CLS & linear head	96.3	88.4	84.8	96.4
Verbalizer & LM head	95.8	86.6	84.6	94.6

Table 4: Comparison between [CLS] label with linear head and verbalizer with LM head on RoBERTa-large.

4.1 P-tuning v2: Across Scales

Table 2 presents P-tuning v2's performances across model scales. In SuperGLUE, performances of Lester et al. (2021) and P-tuning at smaller scales can be quite poor. On the contrary, P-tuning v2 matches the fine-tuning performance in all the tasks at a smaller scale. P-tuning v2 even significantly outperforms fine-tuning on RTE.

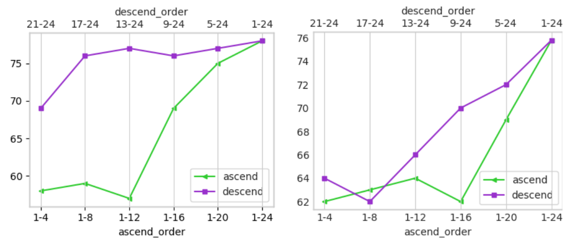
In terms of larger scales (2B to 10B) with GLM (Du et al., 2021), the gap between Lester et al. (2021); Liu et al. (2021) and fine-tuning is gradually narrowed down. On 10B scale, we have a similar observation as Lester et al. (2021) reports, that prompt tuning becomes competitive to fine-tuning. That said, P-tuning v2 is always comparable to fine-tuning at all scales but with only 0.1% task-specific parameters needed comparing to fine-tuning.

4.2 P-tuning v2: Across Tasks

From Table 3, we observe that P-tuning v2 can be generally comparable to fine-tuning on all tasks. P-tuning and Lester et al. (2021) show much poorer performance, especially on QA, which might be the most challenging of the three tasks. We also notice that there are some abnormal results of Lester et al. (2021) and P-tuning on SQuAD 2.0. This is probably because SQuAD 2.0 contains unanswerable questions, which causes optimization challenges for single-layer prompt tuning. Multi-task learning generally brings significant improvements to P-Tuning v2 over most tasks except for QA.

4.3 Ablation Study

Verbalizer with LM head v.s. [CLS] label with linear head. Verbalizer with LM head has been a central component in previous prompt tuning approaches. However, for P-tuning v2 in a supervised setting, it is affordable to tune a linear head with about several thousand parameters. We present our comparison in Table 4, where we keep other hyper-parameters and only change [CLS] label with linear head to verbalizer with LM head. Here, for simplicity, we use "true" and "false" for SST-2, RTE and BoolQ; "true", "false" and "neutral" for CB. Results indicate that there is no significant difference between performances of verbalizer and [CLS].



(a) RTE (b) BoolQ

Figure 3: Ablation study on prompt depth using BERT-large. "[x-y]" refers to the layer-interval we add continuous prompts (e.g., "21-24" means we are add prompts to transformer layers from 21 to 24). Same amount of continuous prompts added to deeper transformer layers (i.e., more close to the output layer) can yield a better performance than those added to beginning layers.

Prompt depth. The main difference between Lester et al. (2021); (Liu et al., 2021) and P-tuning v2 is the multi-layer continuous prompts. To verify its exact influence, given a certain number of k layers to add prompts, we select them in both ascending and descending order to add prompts; for the rest layers, we left them untouched. As shown in Figure 3, with the same amount of parameters (i.e., num of transformer layers to add prompts), adding them in the descending order is always better than in the ascending order. In the RTE case, only adding prompts to layers 17-24 can yield a very close performance to all layers.

5 Conclusions

We present P-tuning v2, a prompt tuning method. Despite its relatively limited technical novelty, it contributes to a novel finding that prompt tuning can be comparable to fine-tuning universally across scales (from 330M to 10B parameters) and tasks. With high accuracy and parameter efficiency, P-Tuning v2 can be a potential alternative for fine-tuning and a strong baseline for future work.

ACKNOWLEDGEMENT

We would like to thank the anonymous reviewers for their suggestions and comments. Jie Tang is supported by the NSFC for Distinguished Young Scholar (61825602) and NSFC (61836013). Kaixuan Ji is supported by Tsinghua University Initiative Scientific Research Program and DCST Student Academic Training Program.

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 89–97, Boston, Massachusetts, USA. Association for Computational Linguistics.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, Michigan. Association for Computational Linguistics.

Xiang Chen, Xin Xie, Ningyu Zhang, Jiahuan Yan, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Adaprompt: Adaptive prompt-based finetuning for relation extraction. *arXiv preprint arXiv:2104.07650*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv e-prints*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2021. All nlp tasks are generation tasks: A general pretraining framework. *arXiv preprint arXiv:2103.10360*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention.

Boseop Kim, HyoungSeok Kim, Sang-Woo Lee, Gichang Lee, Donghyun Kwak, Dong Hyeon Jeon, Sunghyun Park, Sungju Kim, Seonhoon Kim, Dongpil Seo, et al. 2021. What changes can large-scale language models bring? intensive study on hyperclova: Billions-scale korean generative pretrained transformers. *arXiv preprint arXiv:2109.04650*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv:2103.10385*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv e-prints*.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2021. Noisy channel language model prompting for few-shot text classification. *arXiv preprint arXiv:2108.04106*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Timo Schick and Hinrich Schütze. 2020. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *NeurIPS 2019*, pages 3261–3275.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *arXiv e-prints*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv e-prints*.

Hongru Wang, Mingyu Cui, Zimo Zhou, Gabriel Pui Cheong Fung, and Kam-Fai Wong. 2021a. Topicrefine: Joint topic prediction and dialogue response generation for multi-turn end-to-end dialogue system. *arXiv preprint arXiv:2109.05187*.

Shuo Wang, Zhaopeng Tu, Zhixing Tan, Wenxuan Wang, Maosong Sun, and Yang Liu. 2021b. Language models are good translators. *arXiv preprint arXiv:2106.13627*.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Lu Xu, Zhanming Jie, Wei Lu, and Lidong Bing. 2021. Better feature integration for named entity recognition. *arXiv preprint arXiv:2104.05316*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Yanan Zheng, Jing Zhou, Yujie Qian, Ming Ding, Jian Li, Ruslan Salakhutdinov, Jie Tang, Sebastian Ruder, and Zhilin Yang. 2021. Fewnlu: Benchmarking state-of-the-art methods for few-shot natural language understanding. *arXiv preprint arXiv:2109.12742*.

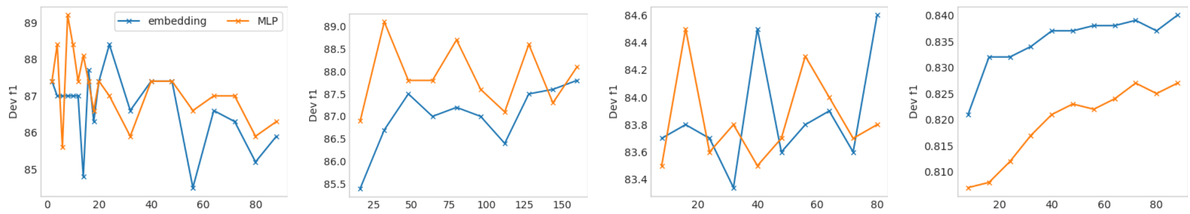
Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall. *arXiv preprint arXiv:2104.05240*.

A Problem Formulation on Sequence Tagging

Name entity recognition (NER). NER aims to predict all spans of words that represent some given classes of entity with a sentence. We adopted CoNLL03 (Sang and De Meulder, 2003), OntoNotes 5.0 (Weischedel et al., 2013) and CoNLL04 (Carreras and Màrquez, 2004). For CoNLL03 and CoNLL04, we trained our model on the standard train-develop-test split. For OntoNotes 5.0, we use the same train, develop, test split as (Xu et al., 2021). All the datasets are labeled in IOB2 format. We use sequence tagging to solve NER tasks by assigning labels marking the beginning and inside some classes of entity. The language models generate a representation for each token, and we use a linear classifier to predict the labels. We use the official scripts to evaluate the results. For the multi-task setting, we combine the training set of the three datasets for pre-training. We use different linear classifiers for each dataset while sharing the continuous prompts.

(Extractive) Question Answering (QA). Extractive QA is designed to extract the answer from the context given the context and a question. We use SQuAD (Rajpurkar et al., 2016) 1.1 and 2.0, in which each answer is within a continuous span of the context. Following tradition, we formulate the problem as sequence tagging by assigning one of the two labels: ‘start’ or ‘end’ to each token and at last selecting the span of the most confident start-end pair as the extracted answer. If the probability of the most confident pair is lower than a threshold, the model will assume the question unanswerable. For the multi-task setting, our training set for pre-training combines the training sets of SQuAD 1.1 and 2.0. When pre-training, we assume that all the questions, regardless of their origin, are possibly unanswerable.

Semantic Role Labeling (SRL). SRL assigns labels to words or phrases in a sentence that indicate their semantic roles in the sentence. We evaluate P-tuning v2 on CoNLL05 (Carreras and Màrquez, 2005) and CoNLL12 (Pradhan et al., 2012). Since a sentence can have multiple verbs, we add the target verb token to the end of each sentence to help recognize which verb is used for prediction. We classify each word with a linear classifier based on the corresponding semantic role representation. For multi-task setting, the pre-train training set is a combina-



(a) NLI: RTE (b) NER: CoNLL04 (c) MQA: BoolQ (d) SRL: CoNLL12

Figure 4: Ablation study on prompt length and reparamerization using RoBERTa-large. The conclusion can be very different given certain NLU task and dataset. (MQA: Multiple-choice QA)

tion of the training set of CoNLL05 (Carreras and Màrquez, 2005), CoNLL12 (Pradhan et al., 2012) and propbank-release (a common extend data used for training SRL). The multi-task training strategy is similar to NER.

B More Ablation Study

Due to the page limit, we present hyper-parameters and architecture designs ablations regarding reparameterization and prompt length in this section.

Embedding v.s. MLP reparameterization. In both prefix-tuning (Li and Liang, 2021) and P-tuning (Liu et al., 2021), authors discover the reparameterization to be useful in improving training speed, robustness and performance. However, we conduct experiments to show that the reparameterization effect is inconsistent across different NLU tasks and datasets.

As shown in Figure 4, in RTE and CoNLL04, MLP reparameterization generally indicates better performance than embedding for almost all prompt lengths. However, in BoolQ, MLP and embedding's results are competitive; in CoNLL12, the embedding consistently outperforms MLP.

Prompt Length. Prompt length is yet another influential hyper-parameter for P-tuning v2, and its optimal value varies from task to task. From Figure 4, we observe that for simple NLU tasks, usually, a shorter prompt is enough for the best performance; for hard sequence tasks, usually, a longer prompt than 100 would be helpful.

We also discover that reparameterization has a close bond with optimal prompt length. For example, in RTE, CoNLL04, and BoolQ, MLP reparameterization achieves its optimal result earlier than embedding. This conclusion may contribute some thoughts on P-tuning's optimization properties.

On Efficiently Acquiring Annotations for Multilingual Models

Joel Ruben Antony Moniz*, Barun Patra*
{jramoniz, barunpatra95}@gmail.com

Matthew R. Gormley
Carnegie Mellon University
mgormley@cs.cmu.edu

Abstract

When tasked with supporting multiple languages for a given problem, two approaches have arisen: training a model for each language with the annotation budget divided equally among them, and training on a high-resource language followed by zero-shot transfer to the remaining languages. In this work, we show that the strategy of joint learning across multiple languages using a single model performs substantially better than the aforementioned alternatives. We also demonstrate that active learning provides additional, complementary benefits. We show that this simple approach enables the model to be data efficient by allowing it to arbitrate its annotation budget to query languages it is less certain on. We illustrate the effectiveness of our proposed method on a diverse set of tasks: a classification task with 4 languages, a sequence tagging task with 4 languages and a dependency parsing task with 5 languages. Our proposed method, whilst simple, substantially outperforms the other viable alternatives for building a model in a multilingual setting under constrained budgets.

1 Introduction

While neural networks have become the de-facto method of tackling NLP tasks, they often require a lot of annotated data to do so. This task of data annotation is especially challenging while building systems aimed at serving numerous languages. Motivated by this, in this paper, we tackle the following problem:

Given the requirement of building systems for an NLP task in a multilingual setting with a fixed annotation budget, how can we efficiently acquire annotations to perform the task well across multiple languages?

The traditional approach to this problem has been building a separate model to serve each language. In this scenario, the annotation budget is split equally for all languages, and a model is trained for each one separately. Recently, another direction that has gained popularity has been leveraging multilingual pre-trained language models (MPLMs) which inherently map multiple languages to a common embedding space (Devlin et al., 2019; Conneau et al., 2020). The popular method for leveraging these models has been leveraging their zero-shot transfer ability: training on an English-only corpus for the task, and then using the models zero-shot for the other languages.

Another orthogonal line of work aimed at building models under a constrained budget has been active learning (AL) (Shen et al., 2018; Ein-Dor et al., 2020). While this has shown to improve annotation efficiency, the predominant approach has been to train one model per language, using the (language specific) model for AL (Shen et al., 2018; Erdmann et al., 2019).

In this work, we show that a single MPLM trained on all languages simultaneously performs much better than training independent models for specific languages for a fixed total annotation budget. Further, while the benefits of using AL in conjunction with MPLMs has been studied for a monolingual setup (Ein-Dor et al., 2020), we show that AL also yields benefits in the multilingual setup. Concretely, we show that an AL acquisition on a single language helps improve zero-shot performance on *all other* languages, regardless of the language of the seed data. Furthermore, we show that AL also yields benefits for our proposed single model scenario. We demonstrate that our results are consistent on 3 different tasks across multiple languages: classification, sequence tagging and dependency parsing. Our approach removes the requirement of maintaining n different models, and uses $1/n^{th}$ the parameters than when independent models are trained. Our analysis reveals that the model arbitrates between different languages based on its performance to form a multilingual curricu-

*Equal Contribution

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 69 - 85
May 22-27, 2022 ©2022 Association for Computational Linguistics

lum.

We release our code at https://github.com/codedecde/SMAL.

2 Related Work

Effective utilization of annotation budgets has been the area of focus for numerous active learning works, showing improvements for different tasks like POS tagging (Ringger et al., 2007), sentiment analysis (Karlos et al., 2012; Li et al., 2013; Brew et al., 2010; Ju and Li, 2012), syntactic parsing (Duong et al., 2018), and named entity recognition (Settles and Craven, 2008; Shen et al., 2018). The focus of most of these works, however, has been on learning for a single language (often English). Prior work on AL that uses a multilingual setup or cross-lingual information sharing and that goes beyond training a separate model for each language has thus been limited. The closest work where multiple languages influence each other's acquisition is that of Qian et al. (2014); however, they still train a separate model for each language.

For transfer to multiple languages, recent advances in building MPLMs (Devlin et al., 2019; Conneau et al., 2020; Liu et al., 2020; Xue et al., 2020) have been extremely effective, especially in zero-shot transfer (Pires et al., 2019; Liu et al., 2020). Ein-Dor et al. (2020) studied the data-effectiveness of these models when used in conjunction with AL, but, as with other AL work, with a single language focus. Finally, Lauscher et al. (2020) studied the effectiveness of the zero-shot setup, showing that adding a few examples to a model trained on English improves performance over zero-shot transfer. However, this assumes the availability of a full English task-specific corpus.

3 Methodology

3.1 Task Specific Models

We use the multilingual-BERT-cased model (mBERT) as the base model for all the tasks. We use the standard training methodology for the tasks: For **classification**, we use a single layer over the [CLS] embedding. For **sequence tagging**, we use a single layer for each word to predict its tag. For **dependency parsing**, we follow Kondratyuk and Straka (2019) and use mBERT embeddings with the graph-based bi-affine attention parser (Dozat and Manning, 2017); refer Appendix A for details.

3.2 Budget Allocation Settings

To understand data acquisition in a multilingual setting, we consider multilingual datasets in the 3 tasks. For each task t, let $\mathcal{L}$ be the set of languages ($n = |\mathcal{L}|$). We then define s_t to be the seed size, b_t to be the total annotation budget and v_t to be total number of annotated validation examples available to t. We compare our proposed Single Model Acquisition (SMA) setup to two baseline settings– Monolingual Acquisition (MonoA) and Multi Model Acquisition (MMA):

MonoA In this setting, the seed data as well as the validation data (s_t, v_t) is acquired from a single language. Further, the entire annotation budget (b_t) is assigned to the same language. We evaluate the test data performance on that language and on the other $n - 1$ languages in a zero-shot setting.

MMA For this setting, we train n individual models, one for each language. Each model starts with a seed of s_t/n, a validation set of v_t/n, and is assigned an acquisition budget of b_t/n. At test time, we evaluate the performance of the model on the language it was trained with.

SMA For this setting, we consider a single model for which both training and acquisition is done on all n languages simultaneously. The seed data and the validation set comprises of a random subset drawn from data corresponding to all languages. The whole of s_t, b_t and v_t are thus assigned to this single model. We compute the performance on the test data of each of the languages.

3.3 Active Learning Acquisition Strategies

The field of active AL tends not to reveal explicit winners—though there is a general consensus that AL does indeed outperform passive learning (Settles, 2009). Thus, we adopt the simplest confidence based strategies to demonstrate their efficacy for each task : Least Confidence (LC) for classification, Maximum Normalized Log Probability (MNLP) (Shen et al., 2018) for sequence tagging, and normalized log probability of decoded tree (NLPDT) (Li et al., 2016) for dependency parsing

Maximum Normalized Log Probability (MNLP) This strategy chooses instances for which the log probability of the model prediction, normalized by sequence length, is the lowest. This AL strategy has been shown to be extremely effective for NER

(Shen et al., 2018) and hence we adopt it in our setting.

Least Confidence (LC) This strategy chooses those instances for which the model confidence corresponding to the predicted class is the least. This acquisition strategy has been commonly applied in classification tasks, and although simple, has been consistently shown to often perform extremely well (Settles, 2009); consequently, we adopt it in our setting.

Normalized Log Probability of the Decoded Tree (NLPDT) This strategy selects the instances with the minimum log probability of the decoded tree generated d^* as generated by the Chu-Liu/Edmonds algorithm (refer A for additional details). Following (Li et al., 2016), we also normalize this score by the number of tokens N [1].

To the best of our knowledge, this is the first work to explore an AL-augmented single model for multiple languages.

4 Experiments

4.1 Dataset Details

Classification We consider Sentiment Analysis, using the Amazon Reviews dataset (Prettenhofer and Stein, 2010). The dataset consists of reviews and their binary sentiments for 4 languages: English (en), French (fr), Japanese (ja), German (de).

Sequence Tagging We choose Named Entity Recognition, and use the CoNLL02/03 datasets (Sang, 2002; Tjong Kim Sang and De Meulder, 2003) with 4 languages: English (en), Spanish (es), German (de) and Dutch (nl), and 4 named entities: Location, Person, Organization and Miscellaneous.

Dependency Parsing We use a subset of treebanks with 5 languages (English (en), Spanish (es), German (de), Dutch (nl), Japanese (ja)) from the full Universal Dependencies v2.3 corpus (Nivre et al., 2018); a total of 11 treebanks.

4.2 Experimental Settings

For each experiment, we run 4 training rounds: one training on initial seed data, followed by 3 acquisition rounds. We set s_t=b_t=v_t in all cases. For classification, we set s_t=300 sentences. For NER and Dependency Parsing, we use s_t=$\sim$$10k$ and s_t=$\sim$$17.5k$ tokens respectively (refer Appendix B). We report accuracy for classification, F1-Score for the NER, and unlabeled and labeled attachment scores (UAS and LAS) for dependency parsing.

For each task, we run the 3 settings (§3.2) across multiple languages. For each setting, we also train an AL model with a task-specific acquisition function (§3.3). In addition, we train both the SMA and MMA with all available data, i.e., we use all data to train one model for all languages and one model per language respectively. We report an average of 5 runs for each experiment. Refer Appendix C for hyperparameters and training details.

5 Results and Analysis

Model Performance Figure 1 shows the performance of NER on Spanish (refer Appendix G for the plots of all other languages and tasks). Although acquiring data independently per language (MMA) performs well, SMA outperforms MMA. Unsurprisingly, MonoA with es performs the best in the category, since it allocates its entire budget to acquiring es data; it thus forms an upper-bound of the model performance. However, SMA outperforms MonoA when its seed language and inference language differ. Finally, AL consistently provides gains over random acquisition.

To analyze the performance across all languages, we present the performance for each round of acquisition, aggregated across all languages for Classification (Figure 2) (refer Appendix G for Dependency Parsing and NER plots). Here, SMA consistently outperforms MMA for every round of acquisition because MMA suffers from a poorly utilized budget, potentially wasting annotation budget on languages where the task is easier. In contrast, SMA improves budget utilization while also benefiting from cross-lingual information. Finally, SMA, by virtue of performing well irrespective of language, consistently outperforms MonoA.

For a concise overview, we present the aggregate metrics across all rounds for each task in Table 1. We observe that SMA does much better compared to its counterparts; both with and without AL. We also observe these models to be extremely data efficient: with AL, a model with access to less than 5% of the data achieves a (relative) performance of around 88% accuracy (for classification), 95.5% F1-score (for NER) and 93.5% LAS (for depen-

[1] We also tried normalizing by N^2, as well as a globally normalized probability of d^* (probability of the tree over all possible valid trees, with the partition function computed using the Matrix Tree Theorem (Koo et al., 2007; Smith and Smith, 2007)), but found both to perform worse.

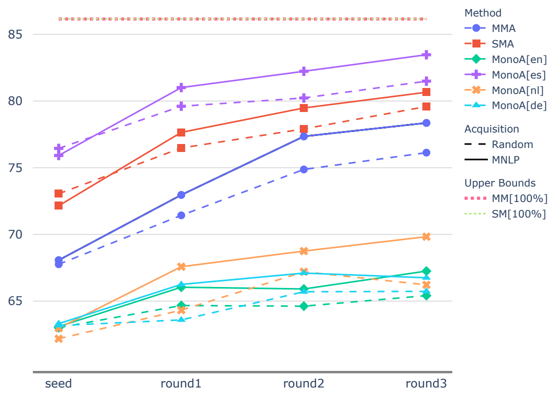


Figure 1: Performance across different rounds for one task (NER) and one language (es). Note that SMA ± AL out-performs MMA ± AL. It also out-performs all MonoA baselines except MonoA[es], which is the language specific upper bound. Here MNLP is the AL method adopted for NER.

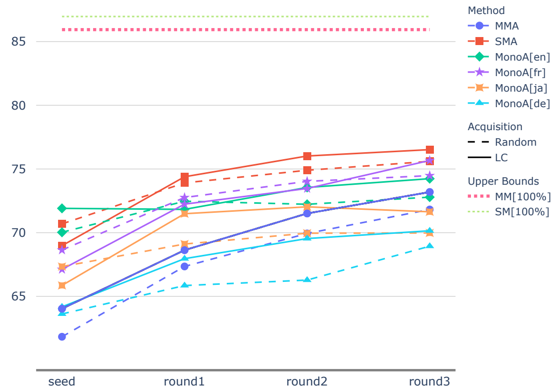


Figure 2: Performance aggregated across all languages for one task (classification) at every round of acquisition. As can be seen, SMA ± AL outperforms all other baselines. Note that SMA and MMA both out-perform MonoA. This is because MonoA does not perform as well when the language is different than that for which data was acquired. Here, LC is the AL method adopted for classification.

dency parsing) when compared to a model trained with all available data (see Table 2 for full data performance). Further, along with its superior performance, SMA also affords substantial parameter savings: requiring only a single model, compared to a number of models linear in n (thereby using $\frac{1}{n^{th}}$ parameters compared to MMA).

Dataset	Metric	AL	MMA	SMA
NER	Span-F1	(-)	75.1	79.1
		(+)	77.3	**80.5**
Classification	Accuracy	(-)	67.7	73.8
		(+)	69.3	**74.0**
Dependency Parsing	UAS	(-)	84.8	86.0
		(+)	84.5	**86.3**
	LAS	(-)	78.0	77.8
		(+)	77.8	**79.7**

Table 1: Average results across all rounds (5%, 10%, 15% and 20% data) and all languages. (+) and (-) indicate with and without AL respectively. Bold highlights best performance for a task.

MM Full vs SM Full To analyze how effectively a single model performs on the languages in question despite using $1/n^{th}$ the parameters, we train a single model on all data and compare it with n language-specific models, where each of the n models has the same number of parameters as the single model; this also serves as an upper-bound for our AL experiments. Table 2 shows that having a single model does not adversely impact performance. A more detailed discussion is present in Appendix D.

Dataset	Metric	MM Full	SM Full
NER	Span-F1	**87.4**	87.2
Classification	Accuracy	86.0	**87.0**
Dependency Parsing	UAS	**91.3**	**91.3**
	LAS	**87.1**	**87.1**

Table 2: Performance with all data for both SM and MM. Here, SM is a single model trained on all languages, while MM represents average performance over all languages of one model per language. The comparable performance indicates that models have enough capacity to represent languages in consideration.

The effectiveness of AL in MonoA We consistently observe AL in the source language improving performance across all languages, irrespective of whether inference is being run for the source language or zero-shot on a different target language, both for NER and classification (Table 3). We hypothesize that the model selects semantically difficult or ambiguous examples that generalize across languages by virtue of mBERT's shared embedding representation. To the best of our knowledge, this work is the first to demonstrate that AL can improve the data efficiency of both classification and NER in a zero-shot inference setup.

In the case of dependency parsing, we observe mixed results when the source and target languages differ. We hypothesize that this is because dependency parsing is a syntactic problem, making it more language specific, and zero-shot inference inherently harder. This is in contrast with both classification and NER, which are more semantic, making hard examples more generalizable across languages. Refer Appendix E for more details.

Dataset	Metric	AL	MonoA				
NER	Source		en	es	nl	de	
	Span-F1	(-)	71.3	64.3	68.8	68.8	
		(+)	**72.1**	64.3	70.8	70.3	
Classification	Source		en	fr	ja	de	
	Acc	(-)	71.9	72.5	69.1	66.2	
		(+)	**72.9**	72.1	70.3	68.0	
Dependency Parsing	Source		en	es	nl	de	ja
	UAS	(-)	76.4	72.9	73.9	72.9	44.3
		(+)	**76.9**	73.0	74.0	73.4	44.2
	LAS	(-)	67.2	62.3	62.8	61.8	31.8
		(+)	**67.5**	62.4	62.7	62.3	30.8

Table 3: Average results across all rounds (5%, 10%, 15% and 20% data) and all languages for MonoAL. Source indicates the language of data acquisition and for all other languages, inference is zero-shot. As can be seen, AL usually helps in the zero-shot setup.

What does SMA+AL acquire? One advantage of the SMA+AL setup is that the model can arbitrate between allocating its acquisition budget across different languages as training progresses. This is in contrast with training one model per language, where the models for languages with a high performance waste the overall budget by acquiring more than necessary, while models on languages where performance isn't as good under-acquire.

To investigate this, for each language and each round, we plot the relative difference (%) between cumulative tokens acquired by the SMA+AL model for that language, and the tokens acquired in expectation if acquisition was done randomly (refer Appendix F for more details). For each language, we also plot the relative performance difference of the language at that round compared to the performance when 100% data is available.

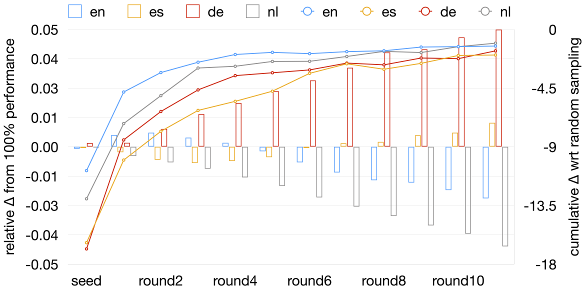


Figure 3: Acquisiton Curriculum for NER. The bars (left y-axis) represent the relative fraction of cumulative tokens acquired per language compared to random sampling. The lines (right y-axis) show the difference of performance of the language when compared to its 100% data performance (MM). Notice that the model tends to favor acquiring data from languages that underperform compared to their 100% counterpart (here, es and de). This in turn helps the model to arbitrate its acquisitions so as to achieve similar performance (relative to 100% performance) across all languages (indicated by the convergence of the line plots).

Figure 3 reveals the added benefit of SMA+AL for data acquisition for NER (refer Appendix F for other tasks): a single model can arbitrate between instances across languages automatically. The model initially acquires data from the high resource language (English). But as the training proceeds, the model favors acquiring data from languages it is uncertain about (Spanish and German). This "multilingual curriculum" thus allows the model to be more effective in its use of the annotation budget. We find SMA+AL eventually achieves a similar relative difference from 100% data performance for all languages consistently across tasks as a consequence.

6 Conclusion

In this work, we consider the problem of efficiently building models that solve a task across multiple languages. We show that, contrary to traditional approaches, a single model arbitrating between multiple languages for data acquisition considerably improves performance in a constrained budget scenario, with AL providing additional benefits.

References

Anthony Brew, Derek Greene, and Pádraig Cunningham. 2010. Using crowdsourcing and active learning to track sentiment in online media. In *ECAI*.

Yoeng-Jin Chu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *ICLR*.

Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip Cohen, and Mark Johnson. 2018. Active learning for deep semantic parsing. In *ACL*.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4).

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *EMNLP*.

Alexander Erdmann, David Joseph Wrisley, Benjamin Allen, Christopher Brown, Sophie Cohen-Bodénès, Micha Elsner, Yukun Feng, Brian Joseph, Béatrice Joyeux-Prunel, and Marie-Catherine de Marneffe. 2019. Practical, efficient, and customizable active learning for named entity recognition in the digital humanities. In *NAACL*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *NLP-OSS*.

Shengfeng Ju and Shoushan Li. 2012. Active learning on sentiment classification by selecting both words and documents. In *CLSW*.

Stamatis Karlos, Nikos Fazakis, Sotiris Kotsiantis, and Kyriakos Sgarbas. 2012. An empirical study of active learning for text classification. *ASSR*, 6(2).

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *EMNLP*.

Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *EMNLP*.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Shoushan Li, Yunxia Xue, Zhongqing Wang, and Guodong Zhou. 2013. Active learning for cross-domain sentiment classification. In *IJCAI*.

Zhenghua Li, Min Zhang, Yue Zhang, Zhanyi Liu, Wenliang Chen, Hua Wu, and Haifeng Wang. 2016. Active learning for dependency parsing with partial annotation. In *ACL*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pretraining for neural machine translation. In *TAACL*.

Joakim Nivre, Mitchell Abrams, Željko Agić, Ahrenberg, et al. 2018. Universal dependencies 2.3. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *ACL*.

Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *ACL*.

Longhua Qian, Haotian Hui, Ya'nan Hu, Guodong Zhou, and Qiaoming Zhu. 2014. Bilingual active learning for relation classification via pseudo parallel corpora. In *ACL*.

Eric Ringger, Peter McClanahan, Robbie Haertel, George Busby, Marc Carmen, James Carroll, Kevin Seppi, and Deryle Lonsdale. 2007. Active learning for part-of-speech tagging: Accelerating corpus annotation. In *LAW*.

Tjong Kim Sang. 2002. Ef: Introduction to the conll-2002 shared task. In *Proceedings of the 6th Conference on Natural Language Learning*.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *EMNLP*.

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep active learning for named entity recognition. In *ICLR*.

David A Smith and Noah A Smith. 2007. Probabilistic models of nonprojective dependency trees. In *EMNLP*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *NAACL*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, et al. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP: System Demo*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

A Task Specific Details

In this section, we elaborate on the task specific adaptations:

Classification: As is common practice, we use a single linear layer over [CLS] embeddings generated by the BERT model to generate logits for the classification task, and the model is trained to minimize the cross-entropy loss.

Sequence Tagging: We apply a linear layer to the word embeddings[2] generated by the BERT model to generate the tag logits, and the model is trained to minimize the negative log-likelihood of the observed tags.

Dependency Parsing: We use a graph-based biaffine attention parser introduced in (Dozat and Manning, 2017). Following (Kondratyuk and Straka, 2019), we use the output of the last BERT layer in place of the embeddings generated by the Bi-LSTM layers. These embeddings are then concatenated with the POS embeddings. A head feed-forward network and a child feed-forward network then generate embeddings for each head and dependant word of a dependency respectively. This is combined with a biaffine attention module to generate a probability distribution for each word to predict its head, as well as a bilinear layer to predict the label for each dependency relationship. Let $\tau_{(i)} = \{(h_{(i,j)}, d_{(i,j)}, l_{(i,j)} | h_{(i,j)} \curvearrowright d_{(i,j)}$ with label $l_{(i,j)}\}$ be the i^{th} gold dependency tree in the dataset. The model is then trained to maximize the log probability of the gold tree as :

$$\max \sum_{i} \sum_{j} \log \left(\mathbb{P}(h_{(i,j)} | d_{(i,j)}) \right) + \log \left(\mathbb{P}(l_{(i,j)} | h_{(i,j)} \curvearrowright d_{(i,j)}) \right) \quad (1)$$

During inference, the best dependency parse is generated by decoding with Chu-Liu/Edmonds algorithm (Chu, 1965; Edmonds, 1967).

For all the models mentioned above, all layers of mBERT are fine-tuned during training.

B Dataset statistics

We report the detailed dataset statistics in Table 4. Note that the seed was chosen to be roughly 5% of the size of the English training data, shown in the rightmost column of the table.

C Experimental Details

Hyperparameters All experiments performed in this paper are averaged over 5 runs. For each experiment, we perform an LR search over (1e-5, 2e-5, 3e-5, 4e-5 and 5e-5), and choose the best LR according to the performance on the appropriate validation (sub)set, as recommended in (Devlin et al., 2019). In all experiments, we set the batch size to 32 and use an Adam (Kingma and Ba, 2015) optimizer. Each round of training is run with a patience of 25 epochs, for at most 75 epochs in total.

Data Preprocessing To avoid out-of-memory issues on the GPU, we pre-process the data so that the examples in the train set of length larger than 175 and with larger than 256 word-pieces are filtered out for the NER. For classification, we simply truncate all instances at 256 word-pieces. We also de-duplicate the train set, to ensure that during all AL acquisition stages, no duplicates are selected at any point.

Code All code used in this work was implemented using Python, PyTorch and AllenNLP (Gardner et al., 2018), using pre-trained models released by HuggingFace (Wolf et al., 2020).

D SM Full vs MM Full Performance

Given that the SMA setup uses $1/n^{th}$ the number of parameters, an interesting question is whether fewer parameters leads to a loss in any expressive power for the single model, which might potentially lead to poorer performance (curse of multilinguality (Conneau et al., 2020)). To answer this question, we train a single model on all data and compare it with n language-specific models, where each of the n models has the same number of parameters as the single model.

From the 100% (rightmost) columns of Table 2, we find that having a single model does not adversely impact performance and these trends hold irrespective of whether all the languages in the task are etymologically close (as in NER) or distant (ja for classification and dependency parsing). This might not be the case when there are a large number of languages, however; investigating how well this observation scales with the number of languages would be an interesting line of future work.

[2] Following (Devlin et al., 2019), for words generating multiple wordpieces, we use the embedding of the first wordpiece.

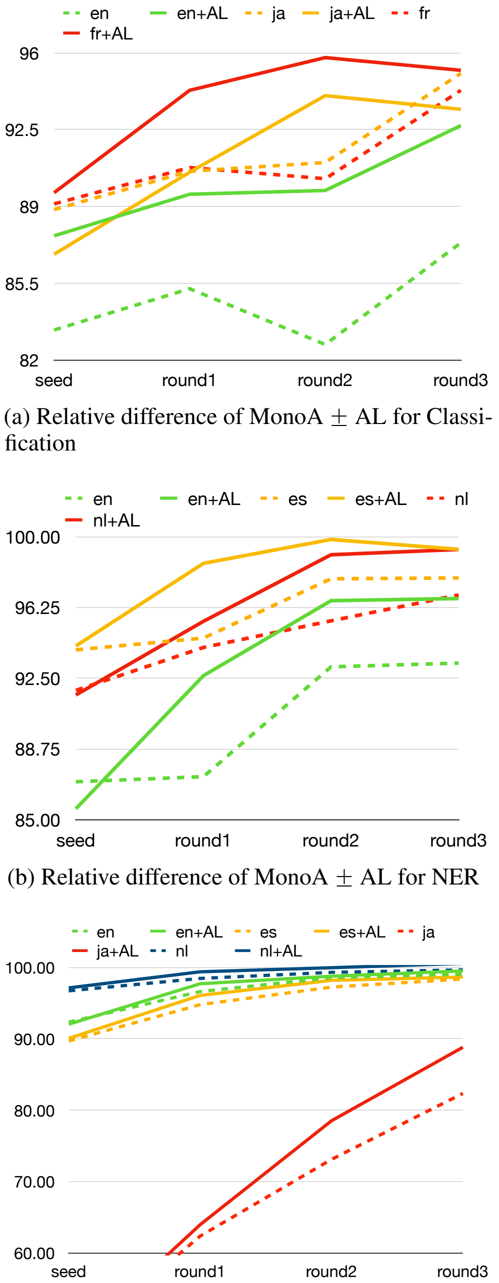
Task	Budget Type	Num Tokens / Instances			AL Details			Num en train
		Train	Val.	Test	Seed	Val.	Budget	
NER	Token	875k	193k	219k	10k	10k	10k	200k
Classification	Instance	19k	5k	24k	300	300	300	6k
Dependency Parsing	Token	1.88M	196k	189k	17.5k	17.5k	17.5k	350k

Table 4: Aggregate statistics of datasets per task.

E Active Learning for the MonoA Setup

An interesting observation from Table 3 is that AL in the source language helps improve performance across all languages, irrespective of whether the inference is being run for the source language in question or zero-shot on a different target language without any training. We observe this to be the case consistently for both the NER and the classification tasks (refer Figure 4), regardless of the source language. We hypothesize that this is because the model selects semantically difficult or ambiguous examples that generalize across languages by virtue of mBERT's shared embedding representation, in contrast with random selection where easy examples the model can already tackle might be selected. We observe this even in the case of etymologically distant languages, such as when the model is trained in English and zero-shot inference is done in Japanese (or vice versa). Thus, the AL selection does not overfit on the specific language in question, instead choosing difficult but generalizable examples.

We observe mixed results for the MonoA setup for dependency parsing: AL improves substantially over Random when the target language and the source language are the same; however, when they differ, the results are mixed. We hypothesize that this discrepancy is a consequence of dependency parsing being a syntactic problem, making it more language specific, in turn making zero-shot an inherently harder problem. This is in contrast with both classification and NER, which are more semantic tasks. Consequently, hard examples for the latter tasks might be more generalizable across languages, resulting in their improved AL performance, when compared with the dependency parsing task.



(a) Relative difference of MonoA ± AL for Classification

(b) Relative difference of MonoA ± AL for NER

(c) Relative difference of MonoA ± AL for Dependency Parsing

Figure 4: Performance of mBERT trained on source (de), as a relative percentage of the performance when all source data was used, in a zero-shot classification setting (es and nl).

F Acquisition Ablation Details and Curriculum

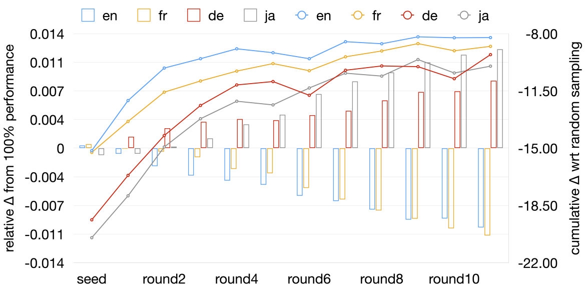


Figure 5: Acquisition curriculum for classification

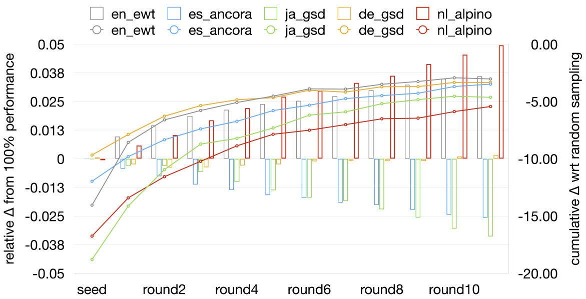


Figure 6: Acquisition Curriculum for dependency parsing. Note that in order to ablate out the effect of different datasets, we only choose the largest dataset for each language.

In this section, we describe the analysis of investigating the acquisitions of SMA+AL in more detail. Let $\alpha_1 \cdots \alpha_n$ be the language specific amount of data present in the entire dataset (i.e $\alpha_i = 0.3$ implies that 30% of the entire dataset (training + unlabeled) is of language i), and let $\beta_{1,1} \cdots \beta_{m,n}$ represent the amount of data acquired for every language at every round (i.e $\beta_{i,j}$ indicates the amount of data acquired by language j at round i). Then, for a task t, for each round i and language j, we plot $\frac{(\sum_{k=1}^{i} \beta_{k,j}) - \alpha_j \dot{b}_t \dot{i}}{\alpha_j \dot{b}_t i}$.

Figures 5 and 6 show the acquisition curriculum. We observe a similar for both the tasks as that for dependency parsing.

G Detailed Results

This section the additional plots as well as the detailed tables and results for all the experiments presented in the paper.

G.1 Per Acquisition Round Performance for Dependency Parsing

Figures 7 and 8 show the UAS and LAS for each round of acquisition for dependency parsing, aggregated across all languages.

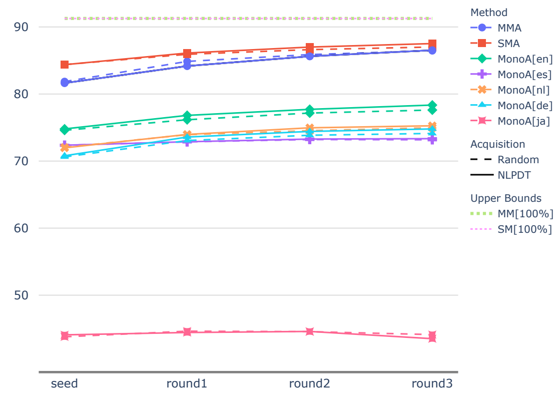


Figure 7: Dependency Parsing: UAS for each round, averaged across all languages

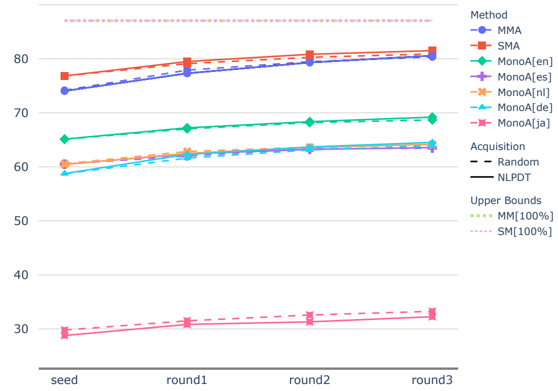


Figure 8: Dependency Parsing: LAS for each round, averaged across all languages

G.2 Per Acquisition Round Performance for NER

Figure 9 shows the F-Score for each round of acquisition for NER, aggregated across all languages.

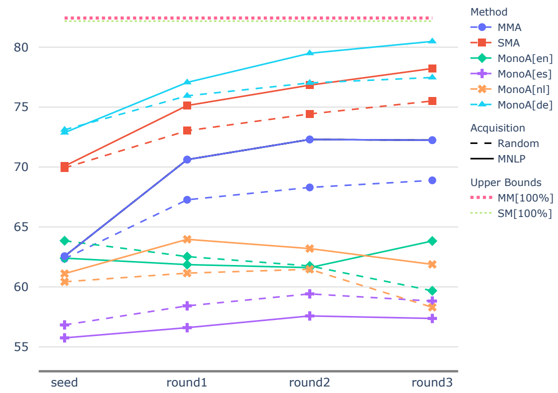


Figure 9: NER: F-Score for each round, averaged across all languages

G.3 Experiments for NER

Tables 5, 6, 7 and 8 show the performance of the different AL settings on English, Spanish, Dutch and German respectively. Each table shows the F-score across 4 acquisition rounds, both with and without MNLP (§3.2).

Acquisition Function		Without MNLP				With MNLP			
Data % Model		**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	86.0 ± 0.6	87.6 ± 0.2	87.8 ± 0.2	88.4 ± 0.4	85.5 ± 0.4	88.4 ± 0.5	89.2 ± 0.2	89.7 ± 0.5
	de	61.3 ± 1.1	61.5 ± 1.5	65.6 ± 2.2	65.7 ± 1.8	60.3 ± 1.6	65.3 ± 3.2	68.1 ± 1.3	68.2 ± 2.3
	es	55.6 ± 1.1	57.2 ± 1.5	56.7 ± 1.5	58.8 ± 1.7	53.7 ± 1.1	56.8 ± 2.7	57.8 ± 3.0	59.5 ± 2.6
	nl	64.8 ± 3.9	64.7 ± 1.1	67.5 ± 0.6	65.7 ± 1.6	67.8 ± 1.6	68.2 ± 2.0	66.4 ± 2.2	66.0 ± 2.4
MMA		81.9 ± 1.4	84.6 ± 0.5	85.3 ± 1.3	86.5 ± 0.7	82.5 ± 0.4	86.1 ± 0.6	87.4 ± 0.6	88.2 ± 0.5
SMA		82.5 ± 0.6	84.8 ± 0.9	85.8 ± 0.4	86.2 ± 0.3	81.9 ± 0.4	86.6 ± 0.6	87.7 ± 0.5	88.4 ± 0.2
MM Full		91.2 ± 0.2							
SM Full		91.2 ± 0.2							

Table 5: Performance (F1-Score) on en for NER

Acquisition Function		Without MNLP				With MNLP			
Data % Model		**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	63.0 ± 1.3	64.7 ± 1.2	64.6 ± 1.1	65.4 ± 1.2	63.1 ± 1.7	66.0 ± 1.2	65.9 ± 1.3	67.3 ± 1.0
	de	63.2 ± 0.5	63.6 ± 0.7	65.7 ± 0.2	65.7 ± 0.6	63.3 ± 1.2	66.3 ± 0.8	67.1 ± 0.7	66.8 ± 0.5
	es	76.5 ± 0.6	79.6 ± 0.7	80.2 ± 0.6	81.5 ± 0.5	75.9 ± 0.5	81.0 ± 0.6	82.2 ± 0.5	83.5 ± 0.3
	nl	62.2 ± 1.0	64.3 ± 1.2	67.2 ± 1.1	66.2 ± 1.4	63.0 ± 1.6	67.6 ± 1.0	68.8 ± 1.4	69.8 ± 1.5
MMA		67.8 ± 1.1	71.4 ± 1.7	74.9 ± 2.4	76.1 ± 2.2	68.1 ± 1.3	73.0 ± 1.9	77.4 ± 0.5	78.4 ± 1.2
SMA		73.1 ± 1.0	76.5 ± 0.7	77.9 ± 0.7	79.6 ± 0.3	72.2 ± 0.9	77.7 ± 0.7	79.5 ± 0.3	80.7 ± 0.5
MM Full		86.2 ± 0.7							
SM Full		86.2 ± 0.5							

Table 6: Performance (F1-Score) on es for NER

Acquisition Function		Without MNLP				With MNLP			
Data % Model		**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	63.9 ± 1.5	62.5 ± 1.0	61.8 ± 1.3	59.7 ± 3.3	62.4 ± 1.4	61.9 ± 1.3	61.6 ± 2.1	63.8 ± 3.3
	de	73.1 ± 0.5	76.9 ± 0.5	77.0 ± 1.2	77.5 ± 0.6	72.9 ± 0.6	77.1 ± 1.1	79.5 ± 0.5	80.5 ± 0.3
	es	56.8 ± 1.1	58.4 ± 1.7	59.4 ± 2.1	58.8 ± 1.6	55.8 ± 2.1	56.6 ± 2.1	57.6 ± 1.3	57.4 ± 1.9
	nl	60.4 ± 1.7	61.2 ± 1.7	61.5 ± 1.1	58.3 ± 3.6	61.1 ± 1.7	64.0 ± 1.3	63.2 ± 2.6	61.9 ± 1.7
MMA		62.4 ± 3.6	67.3 ± 1.2	68.3 ± 1.6	68.9 ± 1.9	62.6 ± 0.6	70.6 ± 1.6	72.3 ± 0.9	72.2 ± 0.8
SMA		69.9 ± 0.8	73.0 ± 0.7	74.4 ± 0.6	75.5 ± 0.6	70.1 ± 0.6	75.1 ± 0.5	76.8 ± 0.2	78.2 ± 0.5
MM Full		82.4 ± 0.5							
SM Full		82.2 ± 0.3							

Table 7: Performance (F1-Score) on de for NER

Acquisition Function		**Without MNLP**				**With MNLP**			
Model	Data %	**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	70.9 ± 0.6	72.1 ± 0.9	71.1 ± 1.2	71.2 ± 1.1	71.3 ± 1.6	71.4 ± 1.1	73.1 ± 1.1	73.4 ± 1.5
	de	69.0 ± 1.8	70.7 ± 0.9	71.8 ± 0.5	72.8 ± 0.6	68.8 ± 2.4	71.8 ± 1.0	74.4 ± 0.8	74.6 ± 0.6
	es	61.6 ± 1.2	62.0 ± 1.5	62.2 ± 1.6	63.5 ± 2.7	62.9 ± 1.3	63.0 ± 1.7	62.8 ± 0.8	62.9 ± 1.6
	nl	82.2 ± 0.5	84.5 ± 0.5	85.2 ± 0.4	85.4 ± 0.6	81.6 ± 1.0	86.8 ± 0.4	88.1 ± 0.4	89.0 ± 0.7
MMA		73.1 ± 1.2	76.4 ± 1.4	77.7 ± 0.5	79.3 ± 1.7	72.0 ± 1.8	78.8 ± 1.4	83.0 ± 0.7	84.7 ± 1.1
SMA		79.8 ± 0.7	82.3 ± 0.3	82.3 ± 0.6	82.8 ± 1.0	79.2 ± 0.8	83.1 ± 0.7	85.1 ± 0.3	86.1 ± 0.2
MM Full		90.0 ± 0.5							
SM Full		89.2 ± 1.2							

Table 8: Performance (F1-Score) on nl for NER

G.4 Experiments for Classification

Tables 9, 10, 11 and 12 show the performance of the different AL settings on English, French, Japanese and German respectively. Each table shows the accuracy across 4 acquisition rounds, both with and without LC (§3.2).

Acquisition Function		**Without LC**				**With LC**			
Model	Data %	**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	74.8 ± 1.2	79.9 ± 0.6	80.7 ± 1.3	81.5 ± 0.2	76.3 ± 1.3	79.9 ± 1.6	81.6 ± 1.4	82.9 ± 1.8
	fr	65.1 ± 5.5	68.9 ± 5.1	72.2 ± 2.9	71.4 ± 4.3	61.3 ± 4.1	64.1 ± 5.6	71.5 ± 5.4	73.3 ± 4.2
	ja	65.7 ± 4.8	66.5 ± 3.8	68.1 ± 3.4	67.1 ± 4.6	63.9 ± 5.9	70.3 ± 3.3	71.7 ± 2.5	70.1 ± 4.6
	de	58.1 ± 1.6	59.4 ± 3.0	57.7 ± 2.6	60.9 ± 4.2	61.1 ± 3.8	62.4 ± 5.1	62.6 ± 6.8	64.6 ± 6.0
MMA		67.1 ± 1.5	71.0 ± 3.7	74.0 ± 4.1	75.1 ± 2.2	67.4 ± 2.9	72.4 ± 3.4	76.0 ± 3.1	76.6 ± 3.6
SMA		73.5 ± 2.1	76.5 ± 0.5	76.7 ± 0.6	77.6 ± 0.8	71.5 ± 3.3	76.9 ± 1.7	78.6 ± 1.4	79.1 ± 0.8
MM Full		86.6 ± 0.3							
SM Full		87.5 ± 0.5							

Table 9: Performance (Accuracy) on en for Sentiment Classification

Acquisition Function		**Without LC**				**With LC**			
Model	Data %	**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	73.1 ± 1.7	75.0 ± 1.4	74.9 ± 2.6	75.7 ± 1.5	74.1 ± 0.6	73.6 ± 2.7	75.5 ± 2.9	76.0 ± 2.2
	fr	75.5 ± 2.6	80.6 ± 0.8	81.7 ± 1.0	83.0 ± 0.8	74.5 ± 1.3	81.4 ± 0.9	82.6 ± 0.5	84.2 ± 0.6
	ja	67.5 ± 4.0	68.7 ± 3.2	68.8 ± 2.4	68.6 ± 2.9	64.9 ± 4.8	71.2 ± 2.8	70.7 ± 1.7	69.8 ± 4.1
	de	64.6 ± 1.2	65.8 ± 3.9	65.4 ± 3.5	68.4 ± 2.3	65.0 ± 2.4	68.4 ± 2.4	69.4 ± 3.2	69.0 ± 4.9
MMA		61.7 ± 3.2	69.9 ± 3.4	73.9 ± 3.0	75.7 ± 2.5	66.0 ± 1.9	71.7 ± 2.1	76.3 ± 0.7	76.6 ± 1.6
SMA		74.6 ± 1.7	77.4 ± 1.2	77.3 ± 0.7	79.2 ± 0.6	72.9 ± 3.1	77.0 ± 1.7	78.4 ± 0.7	79.2 ± 0.8
MM Full		87.8 ± 0.6							
SM Full		89.4 ± 0.4							

Table 10: Performance (Accuracy) on fr for Sentiment Classification

Acquisition Function		Without LC				With LC			
Model	Data %	5%	10%	15%	20%	5%	10%	15%	20%
MonoA	en	65.6 ± 3.2	65.7 ± 3.5	64.6 ± 3.7	64.8 ± 2.04	68.7 ± 3.2	66.0 ± 3.2	67.2 ± 2.7	66.1 ± 3.10
	fr	67.1 ± 2.5	69.9 ± 2.1	70.2 ± 3.2	70.9 ± 1.46	65.7 ± 1.7	71.3 ± 1.3	68.6 ± 3.6	71.4 ± 1.86
	ja	73.0 ± 3.2	75.4 ± 2.4	77.1 ± 1.2	78.8 ± 1.11	71.9 ± 2.4	77.4 ± 0.8	78.6 ± 1.2	79.5 ± 0.68
	de	64.2 ± 1.8	65.5 ± 4.3	65.8 ± 4.6	68.7 ± 1.82	62.7 ± 2.1	65.4 ± 1.6	68.0 ± 3.5	67.5 ± 4.70
MMA		63.6 ± 1.7	68.0 ± 1.8	70.0 ± 1.5	71.8 ± 0.32	63.6 ± 3.0	67.7 ± 1.6	70.3 ± 1.0	70.6 ± 3.14
SMA		65.8 ± 4.3	69.8 ± 2.0	71.1 ± 2.0	71.6 ± 1.01	65.2 ± 2.0	70.1 ± 2.0	72.2 ± 1.1	72.6 ± 1.71
MM Full		83.7 ± 0.3							
SM Full		84.0 ± 0.2							

Table 11: Performance (Accuracy) on ja for Sentiment Classification

Acquisition Function		Without LC				With LC			
Model	Data %	5%	10%	15%	20%	5%	10%	15%	20%
MonoA	en	66.7 ± 2.1	69.4 ± 2.0	68.8 ± 1.9	69.1 ± 2.2	68.5 ± 1.4	67.9 ± 3.3	70.0 ± 2.1	72.0 ± 2.5
	fr	67.0 ± 1.6	71.7 ± 0.8	72.1 ± 1.5	72.6 ± 1.0	67.0 ± 1.6	72.1 ± 0.5	71.3 ± 2.2	73.7 ± 1.1
	ja	63.2 ± 3.1	65.8 ± 2.5	65.9 ± 2.5	65.4 ± 2.4	62.8 ± 3.4	67.1 ± 2.2	67.3 ± 0.8	67.1 ± 3.5
	de	67.5 ± 1.0	72.7 ± 0.8	76.3 ± 0.9	77.8 ± 1.3	67.9 ± 3.8	75.6 ± 1.9	78.2 ± 1.2	79.5 ± 0.7
MMA		55.0 ± 2.1	60.4 ± 1.9	61.9 ± 1.8	64.7 ± 1.4	59.2 ± 1.5	62.8 ± 2.2	63.4 ± 2.7	69.0 ± 2.9
SMA		68.9 ± 1.7	72.0 ± 0.8	74.5 ± 1.0	74.0 ± 1.0	66.3 ± 2.7	73.5 ± 1.0	74.9 ± 1.4	75.2 ± 0.9
MM Full		85.7 ± 0.3							
SM Full		87.0 ± 0.3							

Table 12: Performance (Accuracy) on de for Sentiment Classification

G.5 Experiments for Dependency Parsing

Table 13 compares the performance (LAS and UAS) of the single model trained on all data to the performance of one model trained per language. Table 14 gives the detailed breakdown of each AL setup for each of the dependency parsing datasets, aggregated across all the acquisition rounds.

Model	Metric	en-ewt	en-gum	en-lines	en-partut	es-ancora	es-gsd	de-gsd	nl-alpino	nl-lassysmall	ja-gsd	ja-modern	Avg
MM Full	UAS	92.6	91.3	90.6	93.3	94.1	92.4	89.2	94.4	95.1	95.1	75.9	91.3
	LAS	90.2	88.1	86.2	90.0	91.8	88.9	84.6	92.4	92.4	93.9	58.9	87.1
SM Full	UAS	92.5	91.3	90.8	92.8	94.2	92.6	89.7	94.6	95.0	95.1	75.3	91.3
	LAS	90.1	88.0	86.3	89.8	91.8	89.0	85.2	92.9	92.0	93.8	59.6	87.1

Table 13: Performance on 100% data for Dependency Parsing

Dataset	AL	MonoA										MMA		SMA	
		en		es		de		nl		ja					
		UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
en-ewt	Without NLPDT	89.8†	86.1†	76.1	64.4	77.2	65.9	77.5	66.6	38.7	25.0	85.6	80.4	87.2	82.1
	With NLPDT	90.3†	86.9†	75.5	64.2	77.7	66.2	77.1	65.7	38.1	23.4	85.4	80.5	87.8	83.2
en-gum	Without NLPDT	89.5†	85.3†	77.1	66.0	78.1	68.0	78.0	67.3	37.3	23.8	85.9	80.9	88.0	82.8
	With NLPDT	89.9†	85.9†	76.5	65.9	78.7	68.5	77.9	66.9	37.1	22.5	85.2	80.2	88.0	83.1
en-lines	Without NLPDT	88.9†	84.3†	81.1	69.7	81.5	71.2	82.6	72.6	38.8	24.4	86.0	80.7	87.6	82.4
	With NLPDT	89.1†	84.5†	80.6	69.4	81.8	71.3	82.3	72.1	38.4	22.7	85.4	80.1	88.2	83.0
en-partut	Without NLPDT	91.4†	86.6†	83.4	73.5	82.2	73.0	80.5	72.3	37.6	24.3	87.5	81.5	88.7	82.8
	With NLPDT	90.9†	86.0†	83.3	73.3	82.7	73.3	80.5	72.2	37.5	23.4	86.5	80.6	89.0	83.0
es-ancora	Without NLPDT	82.8	70.8	89.2†	84.6†	81.9	70.1	82.8	69.9	26.7	17.0	85.4	78.6	88.0	81.7
	With NLPDT	82.9	70.1	89.4†	84.7†	82.1	70.6	82.9	69.9	26.4	16.6	84.9	77.9	87.9	81.2
es-gsd	Without NLPDT	83.9	73.4	88.0†	81.5†	83.7	72.1	81.7	69.2	27.4	17.3	84.8	76.6	88.0	81.7
	With NLPDT	83.9	72.9	88.3†	82.0†	84.2	72.7	81.7	68.8	27.1	16.8	84.7	76.9	87.8	81.7
de-gsd	Without NLPDT	84.0	74.4	82.2	71.1	87.2†	81.8†	82.8	71.9	43.9	27.8	84.4	77.8	85.8	79.2
	With NLPDT	84.3	74.6	82.2	71.3	87.6†	82.2†	82.9	71.9	43.4	25.5	84.4	78.0	86.0	79.5
nl-alpino	Without NLPDT	83.0	74.5	82.3	70.1	83.4	74.0	91.8†	88.6†	37.0	21.3	86.1	81.3	87.4	81.4
	With NLPDT	83.2	74.6	82.6	70.3	83.9	74.7	92.1†	89.0†	37.1	21.0	85.8	80.8	87.9	82.0
nl-lassysmall	Without NLPDT	82.1	73.0	82.5	70.8	81.6	71.8	92.6†	88.3†	33.0	20.1	86.9	81.2	88.0	80.9
	With NLPDT	82.2	73.0	82.7	70.8	81.9	72.2	92.8†	88.6†	33.0	18.6	86.4	0.7	88.0	81.1
ja-gsd	Without NLPDT	33.8	17.3	31.6	19.1	33.9	17.1	33.5	13.5	93.1†	91.2†	89.1	85.3	87.4	83.0
	With NLPDT	36.9	18.8	32.8	19.7	34.5	17.6	35.0	14.0	93.7†	91.9†	89.7	86.1	88.2	84.2
ja-modern	Without NLPDT	31.1	14.3	28.5	15.0	31.4	15.2	28.8	10.6	73.7†	57.4†	70.6	53.6	69.8	54.2
	With NLPDT	32.5	15.2	28.8	15.0	32.2	16.4	29.2	10.3	74.0†	57.5†	71.0	54.1	70.0	54.3
Avg.	Without NLPDT	76.4	67.2	72.9	62.3	72.9	61.8	73.9	62.8	44.3	31.8	84.8	78.0	86.0	79.3
	With NLPDT	76.9	67.5	73.0	62.4	73.4	62.3	74.0	62.7	44.2	30.8	84.5	77.8	86.3	79.7

Table 14: Performance on different datasets for dependency parsing. † upper-bounds performance for a particular language (since it assigns the entire budget to that language).

G.6 Language Specific Acquisition Plots

Analogous to Figure 1 in the main paper, each figure in this section presents the performance of the different methods for a specific language and a specific task, at each round of acquisition. The trends observed are fairly consistent: SMA and MMA both do consistently well, with SMA outperforming MMA. MonoA for the specific language does well, but with all other languages performs worse. AL consistently improves performance.

G.6.1 NER

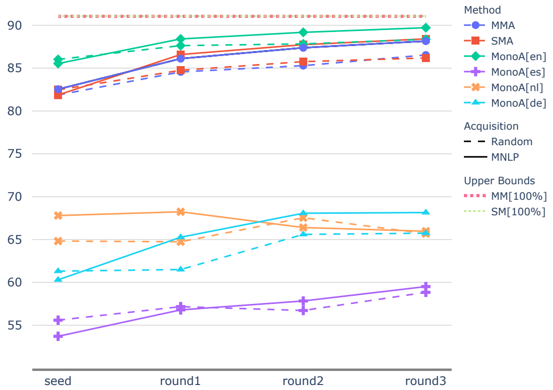


Figure 10: Performance at NER for English (en)

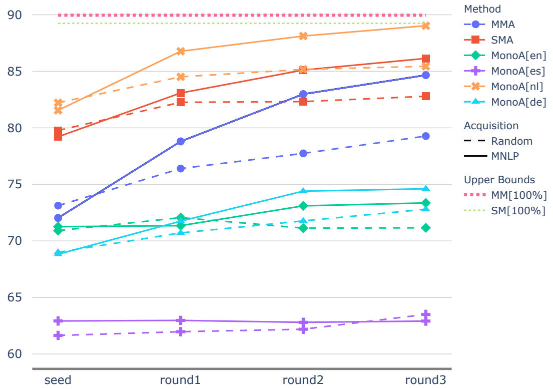


Figure 11: Performance at NER for Dutch (nl)

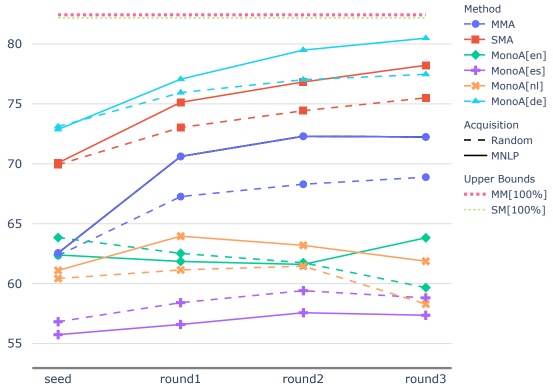


Figure 12: Performance at NER for German (de)

G.6.2 Classification

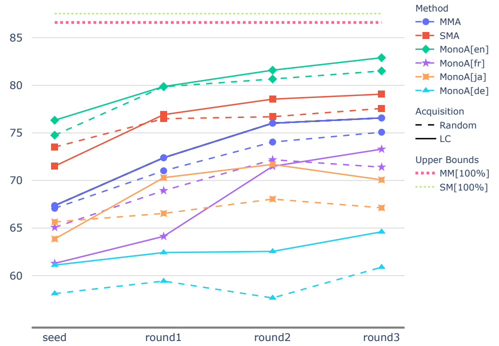


Figure 13: Performance at Classification for English (en)

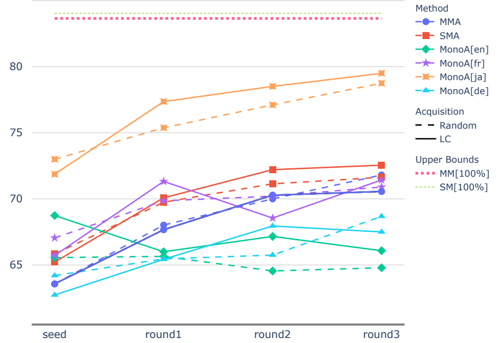


Figure 14: Performance at Classification for Japanese (ja)

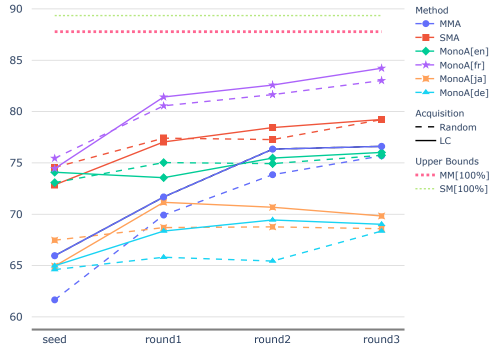


Figure 15: Performance at Classification for French (fr)

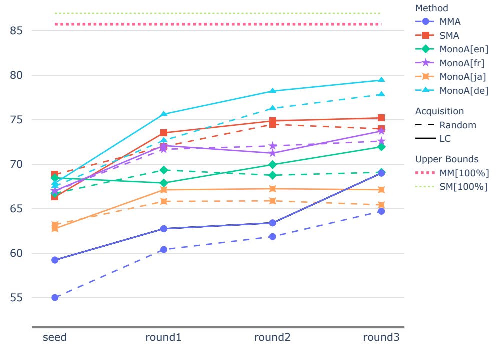


Figure 16: Performance at Classification for German (de)

G.6.3 Dependency Parsing: LAS

For dependency parsing, the MonoA performance of Japanese (MonoA[ja]) is poor on all other languages (Fig. 17, 18, 20, 21, 22, 23, 25, 26), while the performance of all other languages is poor on Japanese (Fig. 19, 24). Consequently, the graphs below have a kink in order to capture this difference in the range of performance of the languages.

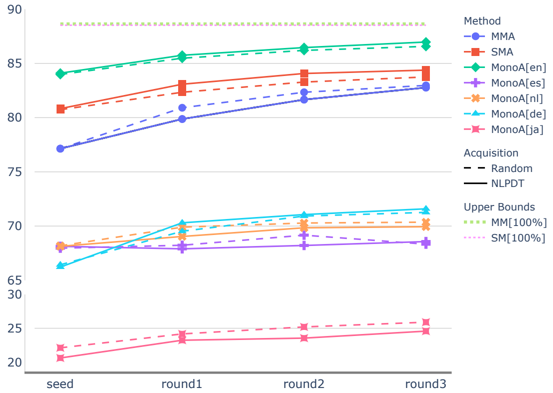


Figure 17: LAS for English (en). Note the kink in the y-axis and the different scales of the two halves.

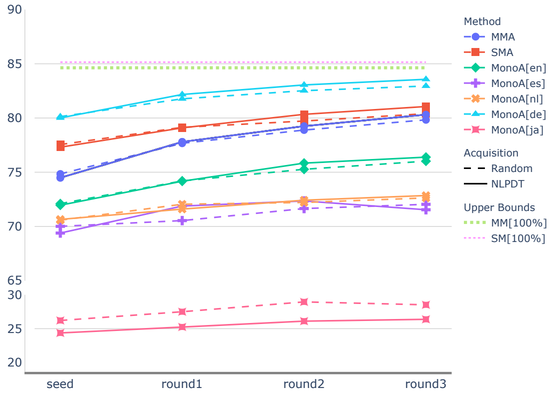


Figure 18: LAS for German (de). Note the kink in the y-axis and the different scales of the two halves.

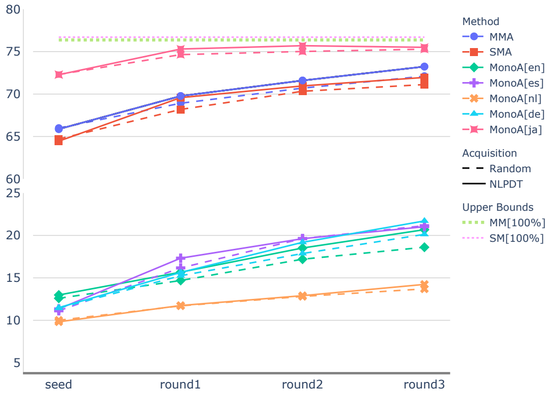


Figure 19: LAS for Japanese (ja). Note the kink in the y-axis and the different scales of the two halves.

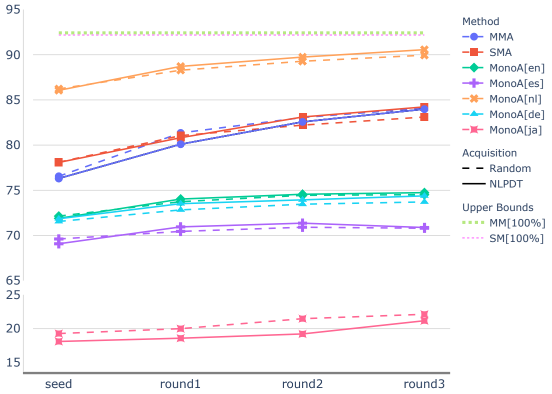


Figure 20: LAS for Dutch (nl). Note the kink in the y-axis and the different scales of the two halves.

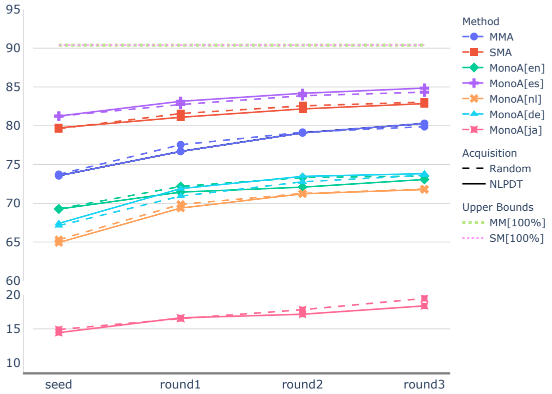


Figure 21: LAS for Spanish (es). Note the kink in the y-axis and the different scales of the two halves.

G.6.4 Dependency Parsing: UAS

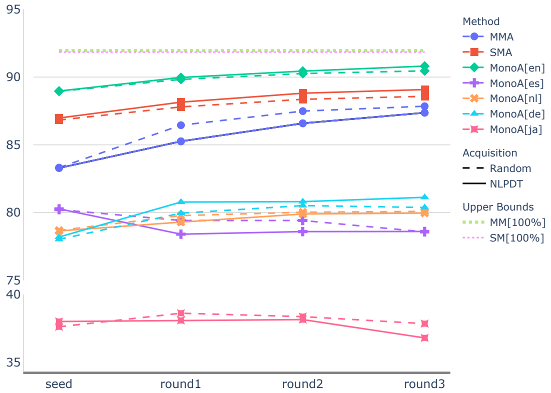


Figure 22: UAS for English (en). Note the kink in the y-axis and the different scales of the two halves.

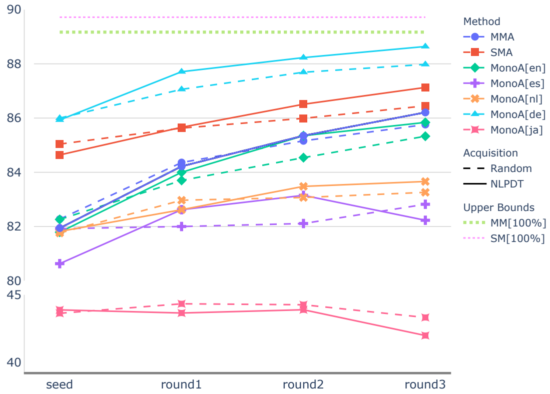


Figure 23: UAS for German (de). Note the kink in the y-axis and the different scales of the two halves.

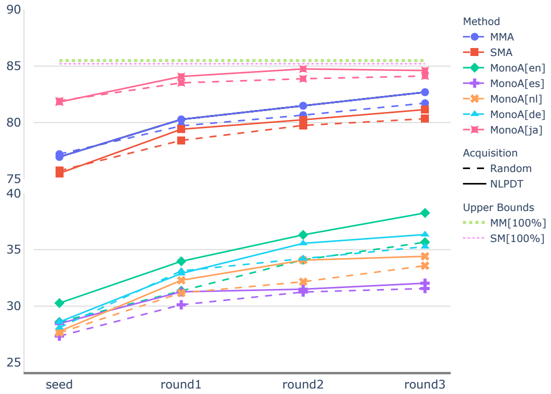


Figure 24: UAS for Japanese (ja). Note the kink in the y-axis and the different scales of the two halves.

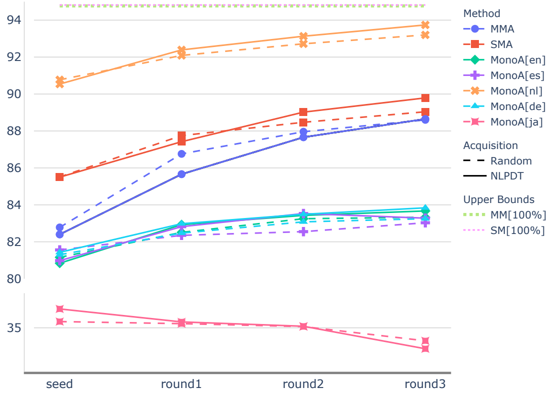


Figure 25: UAS for Dutch (nl). Note the kink in the y-axis and the different scales of the two halves.

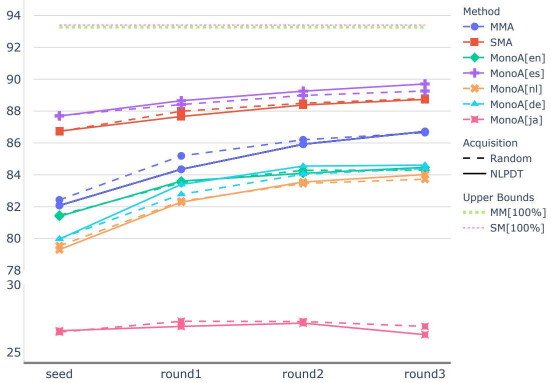


Figure 26: UAS for Spanish (es). Note the kink in the y-axis and the different scales of the two halves.

Automatic Detection of Entity-Manipulated Text Using Factual Knowledge

Ganesh Jawahar[†,‡] **Muhammad Abdul-Mageed**[†] **Laks V. S. Lakshmanan**[†,‡]
[†]Deep Learning & Natural Language Processing Group, [‡]Data Management & Mining Group
The University of British Columbia
ganeshjwhr@gmail.com,{laks,amuham01}@cs.ubc.ca

Abstract

In this work, we focus on the problem of distinguishing a human written news article from a news article that is created by manipulating entities in a human written news article (e.g., replacing entities with factually incorrect entities). Such manipulated articles can mislead the reader by posing as a human written news article. We propose a neural network based detector that detects manipulated news articles by reasoning about the facts mentioned in the article. Our proposed detector exploits factual knowledge via graph convolutional neural network along with the textual information in the news article. We also create challenging datasets for this task by considering various strategies to generate the new replacement entity (e.g., entity generation from GPT-2). In all the settings, our proposed model either matches or outperforms the state-of-the-art detector in terms of accuracy. Our code and data are available at https://github.com/UBC-NLP/manipulated_entity_detection.

1 Introduction

A type of fake news that has received little attention in the research community is manipulated text. Manipulated text is typically created by manipulating a human written news article minimally (e.g., replacing every occurrence of a particular entity, 'Obama' in a news article with another American politician entity). Current fake news detectors that exploit stylometric signals from the text (e.g., choice of specific words to express false statements) are clearly insufficient for distinguishing manipulated text from human written text (Zhou et al., 2019; Schuster et al., 2020) as the style underlying the manipulated text is virtually identical to human writing style. In this work, we focus on this problem of distinguishing manipulated news articles from human written news articles.

Human written text **PubNub**, a startup that develops the infrastructure to power key features in real-time applications (...) has raised $23 million in a series D round of funding from **Hewlett Packard Enterprise (HPE)**, **Relay Ventures**, **Sapphire Ventures**, **Scale Venture Partners**, **Cisco Investments**, **Bosch**, and **Ericsson**.
Manipulated text using GPT-2 **PubNub**, a startup that develops the infrastructure to power key features in real-time applications (...) has raised $23 million in a series D round of funding from **Hewlett Packard Enterprise (HPE)**, **Samsung**, **Sapphire Ventures**, **Scale Venture Partners**, **Cisco Investments**, **Bosch**, and **Ericsson**.

Table 1: Example human written and manipulated text. Named entities of organization type are shown in **green**. Manipulated entities are shown in **orange**.

We consider a particular type of text manipulation — entity perturbation (Zhou et al., 2019), where a manipulated news article is created by modifying a fixed number of entities in a human written news article (e.g., replacing them with entities generated from a text generative model). E.g., in Table 1, to mislead humans, the entity 'Relay Ventures' can be replaced by 'Samsung' (a candidate replacement entity generated by the generative pretraining-2 model (GPT-2) (Radford et al., 2019)), which is locally consistent as some of the other companies in the original text are also into device manufacturing.

To distinguish a manipulated news article from the original human written news article, we propose a neural network based detector that jointly utilizes the textual information along with the the factual knowledge explicitly by building entity-relation graphs which capture the relationship between different entities present in the news article. The factual knowledge is encoded by a graph convolutional neural network (Kipf and Welling, 2017) that captures the interactions between different entities and relations, which we hypothesize, carries discriminatory signals for the manipulated text detection task.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 86 - 93
May 22-27, 2022 ©2022 Association for Computational Linguistics

Our major contributions include: (i) a detector that exploits factual knowledge to overcome the limitations of relying only on stylometric signals, (ii) an approach to generate challenging manipulated news article dataset using GPT-2, and (iii) a collection of challenging datasets by considering various strategies to generate the replacement entity.

2 Background and Related Work

The manipulated text detection task is related to diverse research areas such as fake news detection, natural language understanding, and knowledge bases.

Fake news detection. Research on Fake news detection typically deals with challenges such as understanding the news content (Schuster et al., 2020), claim verification (Thorne and Vlachos, 2018), verifying the credibility of the source (Castillo et al., 2011), and exploiting fake news propagation patterns (Vosoughi et al., 2018). Our work is primarily focused on detecting fake news in the form of manipulated text, by understanding the news content. In the traditional problem setting, both fake and real news is assumed to be written by a human (Shu et al., 2017; Oshikawa et al., 2020). Since humans tend to make stylistic choices (e.g., choosing some specific language for writing false statements), the fake news detector can perform reasonably on the task by picking up on these stylometric signals. One can also create fake news by manipulating a human written news article minimally. Such manipulations include: entity perturbation (e.g., '12 people were injured in the shooting' to '24 people were killed in the shooting') (Zhou et al., 2019), subject-object exchange (e.g., 'A gangster was shot by the police' to 'A policeman was shot by the gangster') (Zhou et al., 2019), and adding/deleting negations (e.g., 'Trump doesn't like Obamacare' to 'Trump likes Obamacare') (Schuster et al., 2020). These manipulations do not typically affect the style and hence stylometric signals alone cannot help in building accurate manipulated text detection models (Zhou et al., 2019; Schuster et al., 2020).

Natural language understanding. Pre-trained language models such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) achieve strong performance in diverse NLP tasks. Specifically, RoBERTa is the state-of-the-art detector when finetuned for detection of synthetic text (Solaiman et al., 2019; Jawahar et al., 2020). These models can also capture implicit world knowledge (e.g., Paris is the capital of France) that occurs frequently in the text (Petroni et al., 2019). However, it is insufficient for solving our task (Schuster et al., 2020), as it is limited to frequent patterns.

Knowledge bases (KBs). Knowledge bases (e.g., YAGO (Tanon et al., 2020)) containing typically a collection of facts (e.g., subject-relation-object triples), provide specialized knowledge for downstream NLP tasks (e.g., question answering (Banerjee and Baral, 2020)). One can integrate such symbolic knowledge into pre-trained language models during pre-training (Zhang et al., 2019) and finetuning (Liu et al. (2020); Zhong et al. (2020), which we follow in this work).

3 Manipulated Text Creation

In this work, we focus on a particular type of manipulation — entity perturbation (Zhou et al., 2019), where all occurrences of a fixed number of randomly picked entities from a human written news article are replaced with different replacement entities. We replace named entities of three types: person, organization and location (recognized using spaCy's named entity recognizer (NER) (Honnibal et al., 2020)). We ensure the replacement (new) entity belongs to the same type as the original (old) entity. We create challenging manipulated text datasets by considering various strategies to identify the new replacement entity: random most frequent entity (pick randomly from among the top 5000 entities), random least frequent entity (pick randomly from the bottom 5000 entities), and entity generated by GPT-2. Sample manipulated entities obtained from different replacement strategies are shown in Table 2.

Entity replacement strategy		
Random least	**Random most**	**GPT-2 generated**
Inverkeithing High School	Tribune	U.S.
Mark Forman	East Jerusalem	Canada
Netgear	Englishman	Microsoft
Bangalore North	Jason Aldean	Donald Trump
Mackintosh	UFA	BBC

Table 2: Sample manipulated entities

GPT-2 generated entity replacement. Strategies that randomly identify the replacement entity ignore the context provided by the news article. For example, in news portion (1), a random replacement entity for 'Relay Ventures' can be 'Salesforce'. However, it is likely locally inconsistent as 'Salesforce' is not into device manufacturing unlike many other co-occurring companies in the original text. We propose a novel approach that makes use of the state-of-the-art text generative model GPT-2 to pick replacement entities that are locally consistent. Revisiting the news portion (1), let the randomly selected entity to be replaced be 'Relay Ventures'. We treat the fragment of text from the beginning of the article up to the tokens before the first occurrence of the target entity ('Relay Ventures') as the prompt. We provide this prompt to GPT-2, which can then generate the next few tokens. We call the generated token sequence a candidate replacement entity if the sequence starts with an entity (e.g., 'Samsung') of same type as the target entity ('Relay Ventures') and has no string overlap with the target entity. If the constraints are not met, we ask GPT-2 to create the generated sequence again up to a maximum of 10 attempts. The candidate replacement entity thus obtained will be used to replace all occurrences of the target entity. For the news portion (1), the candidate replacement entity generated by GPT-2 is 'Samsung', which is locally consistent: similar to other companies in the original text, Samsung manufactures devices.

4 Manipulated Text Detection

The goal of this work is to build a detector that distinguishes manipulated news article from human written news article with high accuracy. In prior work, Zhou et al. (2019) conclude that the manipulated article can possibly be detected by checking the facts underlying the article with knowledge bases and Schuster et al. (2020) show that humans can identify the manipulated text well when they are allowed to consult external sources (e.g., internet). Building on these findings, we hypothesize that *factual knowledge underlying the news article can provide discriminatory signals for manipulated text detection.* To this end, we embody the RoBERTa detector with explicit factual knowledge so that the detector can reason about facts present in the news article, whose details we discuss next.

Factual knowledge. For factual knowledge, we leverage a variant of YAGO 4 KB (Tanon et al., 2020) that contains only instances that have an English Wikipedia article. We then extract the facts in a given document by first identifying all the entities present in the document using spaCy's NER. For each target entity, we grab all the triples in the KB where the subject matches with the target entity at surface level. These triples can be seen as the first hop neighbors of the target entity in the KB. For a given document, the set of triples collected over all identified entities is used to build the corresponding factual graph. A node can be an entity or a relation. A directed edge is added between subject and relation, as well as relation and object. This factual graph contains rich factual information about entities present in the document, which can be exploited to reason about facts mentioned in the article for correctness.

Integrating factual knowledge with RoBERTa. Our proposed detector is an integration of the RoBERTa model with factual knowledge. This allows the detector to reason about facts mentioned in the article. To embed the factual knowledge, we employ graph convolutional networks (GCNs) (Kipf and Welling, 2017), where we stack l GCN layers and the definition of the hidden representation of each node v of the factual graph as layer $k+1$, in a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$:

$$\mathbf{h}_v^{k+1} = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} \mathcal{W}^k h_u^k + b^k\right), \quad \forall v \in \mathcal{V}, \tag{1}$$

where $\mathcal{W}^k$, b^k, h_u^k, $\mathcal{N}(v)$ correspond to layer specific model weights, biases, node representation, and neighbors of v in $\mathcal{G}$ respectively. Note that h_u^1 denotes the initial node features, which can be initialized randomly or using a pre-trained entity embedding such as Wikipedia2vec (Yamada and Shindo, 2019).

Detector prediction. The factual knowledge about entities present in the article is captured in the node embeddings (h_u^l) corresponding to the last layer l of the GCN model. The textual knowledge corresponding to the document can be obtained from the last layer representation (r_{CLS}^d) of the RoBERTa model corresponding to the first token ('[CLS]', special classification token) of the RoBERTa input. We combine the factual and the textual knowledge by simply averaging all the GCN's entity embeddings and concatenating the entity average with the RoBERTa's document embedding. Thus, the unnormalized prediction probabilities ($mf(d)$) of our detector for the document d can be given by:

$$\mathbf{mf}(d) = \mathcal{W}_{mtd}\left[r^d_{[CLS]}; \sum_{e\in entities(d)} h_e^l\right] + b_{mtd}, \tag{2}$$

where $[;]$ corresponds to the concatenation operation and $\mathcal{W}_{mtd}$, b_{mtd} correspond to the affine transformation specific model parameters for manipu-

Entity replacement strategy	**Random least frequent entity replacement**			**Random most frequent entity replacement**			**GPT-2 generated entity replacement**		
Maximum no. of entity replacements	1	2	3	1	2	3	1	2	3
Manipulated Article Detection Task									
(1) Overall Accuracy									
RoBERTa	67.09	78.37	84.26	65.56	76.86	83.93	67.09	74.12	78.79
Ours (w/o Entity Identification Objective)	67.25	78.36	**84.59**	66.99*	77.98*	83.86	**67.16**	73.84	**79.11**
Ours	**68.25***	**78.99**	83.84	**67.21***	**78.26***	**84.39**	65.84	**74.80**	**79.05**
Manipulated Entity Identification Task									
(1) Overall Precision - Ours	49.99	50.02	50.08	49.94	50.00	49.83	49.49	48.52	48.71
(2) Overall Recall - Ours	38.56	55.11	65.11	48.20	50.04	47.71	45.82	46.76	45.67
(3) Overall F-Score - Ours	42.29	46.50	46.12	46.07	47.79	46.83	44.82	47.42	44.92
(4) Manipulated Entity - Precision - Ours	81.06	91.76	84.14	84.71	88.06	86.06	85.59	85.91	73.80
(5) Manipulated Entity - Recall - Ours	0.00	3.70	12.12	6.08	4.63	14.03	9.14	1.64	12.50
(6) Manipulated Entity - F-Score - Ours	0.00	7.11	21.19	11.35	8.80	24.13	16.52	3.22	21.38

Table 3: Evaluation performance (%) for different maximum number of entity replacements across different replacement strategies. **Bolded** refers to the best results for each dataset. Note that the state-of-the-art detector cannot identify manipulated entities present in the document. For the manipulated article detection task, statistically significant overall accuracy results obtained using bootstrap test with $p < 0.01$ are marked using asterisk (*).

lated text detection. The output from $mf(d)$ passes through dropout followed by ReLU layer.

Identifying manipulated entities. To enable humans to understand our detector's decision and perform further investigation, we introduce a subtask for the detector, namely identify the manipulated entities among different entities present in the document. For this subtask, we build on the entity representations output by the last layer of the GCN model. The unnormalized class prediction probabilities ($ef(v)$) for a given entity v from the article can be given by:

$$\mathbf{ef}(v) = Dropout\left(ReLU\left(\mathcal{W}_{ec}h_v^l + b_{ec}\right)\right), \qquad (3)$$

where h_v^l denotes the hidden representation at last layer l for the entity v, and $\mathcal{W}_{ec}$, b_{ec} correspond to the affine transformation specific model parameters for entity classification. The overall objective function of the proposed detector can be given by:

$$\min_{\theta} \sum_{i=1}^{n} \left[\mathcal{L}(s(mf(x_i)), y_i) + \sum_{e \in entities(x_i)} \mathcal{L}(s(ef(e)), y^e) \right]. \qquad (4)$$

where $\mathcal{L}$, mf, and s resp. correspond to the function that computes the negative log-probability of the correct label, detection prediction function, and softmax function. y^e denotes the entity manipulation class label, which is 1 if the entity e is manipulated, and 0 otherwise. y_i denotes the article manipulation class label, which is 1 if at least one entity in article i is manipulated, and 0 otherwise.

5 Experiments and Results

Dataset and Detector Settings. The human written news articles used in our study are taken from the RealNews dataset (Zellers et al., 2019), which contains 5000, 2000, and 8000 news articles in the training, validation, and test set respectively. We randomly pick half of the news articles in each set for human written news article category and the rest in each set for manipulation based on the chosen replacement strategy. We also create three different datasets for each replacement strategy by varying the maximum number of entities to be manipulated from 1 to 3. Detailed statistics of the proposed datasets is in A.1. The hyperparameter search space for all detectors is offered in A.2.

Hardest detection task. Table 3 presents the detection accuracy results. We observe that the most challenging dataset for the state-of-the-art detector is surprisingly from random most frequent entity replacement strategy with exactly one entity replacement. The random strategies fail to create a challenging dataset with high (e.g., 3) number of entity replacements, which indicates that the detection task becomes easier with increase in the number of locally inconsistent entities. Nevertheless, our proposed GPT-2 based entity replacement strategy keeps the detection task harder even for large number of replacements, thanks to the ability of the strategy to generate locally consistent entities mostly. Regardless of the replacement strategies, the detection performance of all the detectors increases with the increase in the number of entities that are manipulated in a document, that is, more the manipulations in a document, the easier the detection task. This result is similar to previous research which performs manipulation by adding/deleting negations in news articles (Schuster et al., 2020). A fake news propagator can thus

Entity replacement strategy	**Random least frequent entity replacement**			**Random most frequent entity replacement**			**GPT-2 generated entity replacement**		
Maximum no. of entity replacements	1	2	3	1	2	3	1	2	3
Test set size (Percent)	3,797 (47.5)	3,625 (45.3)	3,447 (43.1)	3,288 (41.1)	2,660 (33.2)	2,207 (27.6)	3,302 (41.3)	2,737 (34.2)	2,359 (29.5)
RoBERTa	48.17	68.69	77.81	45.62	66.32	74.94	51.97	**66.97**	**74.95**
Ours (w/o Entity Identification Objective)	47.20	65.19	**78.76**	51.55	68.20	**75.44**	56.27	66.68	72.11
Ours	**52.04**	**68.99**	75.98	**54.65**	**68.38**	72.81	**62.11**	66.53	71.22

Table 4: Manipulated article detection performance (%) for different maximum number of entity replacements across different replacement strategies on a subset of our test set. This text subset contains manipulated articles with all the manipulated entities absent in the knowledge base. **Bolded** refers to best results for each dataset.

manipulate exactly one entity in the news article to make the detection task harder.

Detector performance. Nevertheless, our proposed detector performs similarly to or outperforms the state-of-the-art detector on all replacement strategies across different numbers of entity replacements. This result validates our hypothesis that leveraging both factual and textual knowledge can improve detection performance, overcoming the limitations of relying only on textual knowledge. Improvements of our proposed detector on the GPT-2 generated entity manipulation task are not significantly high due to sizeable increase in manipulated entities absent in the knowledge base (e.g., ∼50%, see last three rows in Table 6).

Entity identification performance. Our proposed detector is equipped to identify entities that are manipulated in a news article. This task is harder due to the imbalanced nature of the task as most of the entities present in the news article are not manipulated. As shown in Table 3, our proposed detector achieves high precision ($\geq$ 70%) in identifying manipulated entities, which makes our detector appealing for applications that favor precision. The recall is very low ($<$ 15%), which indicates the difficulty of the task. We also experiment with a baseline RoBERTa model trained at the token level to identify spans of manipulated entities. However, the model seems overwhelmed by the majority class (token not part of the manipulated entity span) and predicts all the tokens to belong to the majority class. We believe there is a lot of room for improvement in this subtask.

Detecting articles with unknown manipulated entities. Table 4 shows performance of the detector on manipulated articles when all the manipulated entities are not present in the knowledge base. We observe that our proposed detector can rely on the relations corresponding to the non-manipulated entities and pretrained textual representations to outperform, or at least be on par with, the RoBERTa model.

Repl. strategy / # replacements	1	2	3
Random least frequent	93.67	95.06	95.05
Random most frequent	93.75	93.37	93.79
GPT-2 generated	95.1	93.35	94.88

Table 5: Quality gap - Human vs. Manipulated text

Quality gap between human and manipulated text. Table 5 shows how the quality of the manipulated text changes with respect to human written text across different replacement strategies, for different numbers of replacements. We utilize MAUVE (Pillutla et al., 2021), a metric to measure the closeness of machine generated text to human language based on divergence frontiers. Since the proposed manipulations touch only limited spans (i.e., entities) in the entire document, the overall quality of the manipulated text does not change much with more replacements.

6 Conclusion

We presented the first principled approach for developing a model that can detect entity-manipulated text articles. In addition to textual information, our proposed detector exploits explicit factual knowledge from a knowledge base to overcome the limitations of relying only on stylometric signals. We constructed challenging manipulated datasets by considering various entity replacement strategies, including with random selection and GPT-2 generation. On all the experimental settings, our proposed model outperforms (or is at least on par with) the baseline detector in overall detection accuracy. Our results show that manipulated text detection remains challenging. We hope that our work will trigger further research on this important but relatively understudied subfield of fake news detection.

Acknowledgements

We gratefully acknowledge support from the Natural Sciences and Engineering Research Council of Canada (NSERC; RGPIN-2018-04267), the Social Sciences and Humanities Research Council of Canada (SSHRC; 435-2018-0576), Canadian Foundation for Innovation (CFI; 37771), Compute Canada (CC),[1] UBC ARC-Sockeye,[2] and Advanced Micro Devices, Inc. (AMD). Any opinions, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSERC, SSHRC, CFI, CC, ARC-Sockeye, or AMD. We also thank Ayushi Dalmia for proofreading and helpful discussions.

References

Pratyay Banerjee and Chitta Baral. 2020. Self-supervised knowledge triplet learning for zero-shot question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 151–162, Online. Association for Computational Linguistics.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, page 675–684, New York, NY, USA. Association for Computing Machinery.

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

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

Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks Lakshmanan, V.S. 2020. Automatic detection of machine generated text: A critical survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2296–2309, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: Enabling language representation with knowledge graph. In *Proceedings of AAAI 2020*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Ray Oshikawa, Jing Qian, and William Yang Wang. 2020. A survey on natural language processing for fake news detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6086–6093, Marseille, France. European Language Resources Association.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. MAUVE: Measuring the gap between neural text and human text using divergence frontiers. In *Advances in Neural Information Processing Systems*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. `https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf`.

Tal Schuster, Roei Schuster, Darsh J. Shah, and Regina Barzilay. 2020. The Limitations of Stylometry for Detecting Machine-Generated Fake News. *Computational Linguistics*.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *SIGKDD Explor. Newsl.*, 19(1):22–36.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release Strategies and the Social Impacts of Language Models. *CoRR*, abs/1908.09203.

Thomas Pellissier Tanon, Gerhard Weikum, and Fabian M. Suchanek. 2020. YAGO 4: A reasonable knowledge base. In *The Semantic Web*, volume 12123 of *Lecture Notes in Computer Science*, pages 583–596.

[1] https://www.computecanada.ca

[2] https://arc.ubc.ca/ubc-arc-sockeye

James Thorne and Andreas Vlachos. 2018. Automated fact checking: Task formulations, methods and future directions. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3346–3359, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science*, 359(6380):1146–1151.

Ikuya Yamada and Hiroyuki Shindo. 2019. Neural attentive bag-of-entities model for text classification. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 563–573. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, volume 32, pages 9054–9065. Curran Associates, Inc.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

Wanjun Zhong, Duyu Tang, Zenan Xu, Ruize Wang, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Neural deepfake detection with factual structure of text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2461–2470, Online. Association for Computational Linguistics.

Zhixuan Zhou, Huankang Guan, Meghana Moorthy Bhat, and Justin Hsu. 2019. Fake news detection via NLP is vulnerable to adversarial attacks. In *Proceedings of the 11th International Conference on Agents and Artificial Intelligence, ICAART 2019, Volume 2, Prague, Czech Republic, February 19-21, 2019*, pages 794–800.

A Appendices

A.1 Summary Statistics of Proposed Datasets.

Table 6 displays the statistics of proposed datasets.

A.2 Hyperparameter Search Space for All Detectors

Table 7 displays the search space for hyperparameters used to tune all the detectors.

Name	**Random least frequent entity replacement**			**Random most frequent entity replacement**			**GPT-2 generated entity replacement**		
Maximum no. of entity replacements	1	2	3	1	2	3	1	2	3
Dataset Size									
Train	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000
Validation	2,000	2,000	2,000	2,000	2,000	2,000	2,000	2,000	2,000
Test	8,000	8,000	8,000	8,000	8,000	8,000	8,000	8,000	8,000
Average Length (# words)									
Train	604	604	605	603	603	603	603	613	614
Validation	595	595	596	594	594	594	607	598	599
Test	597	597	597	596	596	596	598	598	601
% Documents with Person Entities									
Train	97.92	98.00	97.96	97.74	97.84	98.00	97.22	97.60	97.82
Validation	98.65	98.65	98.85	98.55	98.65	98.50	97.80	98.00	98.30
Test	97.86	98.04	98.16	97.92	97.91	97.95	97.45	97.49	97.76
% Documents with Organization Entities									
Train	99.14	99.12	99.10	99.20	99.26	99.12	99.04	99.10	99.14
Validation	99.35	99.35	99.30	99.35	99.35	99.40	99.20	99.50	99.25
Test	99.28	99.20	99.17	99.24	99.12	99.17	99.06	99.05	99.11
% Documents with Location Entities									
Train	90.44	90.16	89.84	90.70	90.70	91.00	90.70	91.34	91.88
Validation	90.40	89.90	89.75	90.55	90.55	90.80	90.80	91.05	91.90
Test	90.69	90.28	89.91	90.83	90.64	90.66	90.95	91.05	91.62
Average % Entity Coverage by YAGO-4									
Train	9.78	9.63	9.46	9.97	10.01	10.03	10.01	10.03	10.01
Validation	9.80	9.62	9.51	9.98	10.03	10.10	9.68	10.02	10.15
Test	9.85	9.70	9.54	10.05	10.07	10.09	10.05	10.01	10.10
Avg. % Known Ents. post Manipulation									
Train	6.94	9.26	11.07	30.28	26.33	28.35	60.85	54.26	51.83
Validation	11.97	9.07	10.16	26.76	23.89	27.18	48.72	49.26	48.68
Test	7.68	8.99	9.03	26.13	27.15	25.76	48.85	52.72	51.51

Table 6: Summary statistics of proposed datasets.

Hyperparameter Name	**Hyperparameter Values**
RoBERTa model variant	Large
Minimum frequency of node (i.e., entity)	{10}
Batch size	{8}
Initial learning rate	{1e-5, 2e-5, 3e-5}
Epochs	{10}
Number of warmup steps	{10%}
Node intialization	{Wikipedia2vec}
Node embedding size	{100, 300}
Number of GCN layers	{1, 2}

Table 7: Hyperparameter search space for all detectors.

Does BERT Know that the IS-A Relation Is Transitive?

Ruixi Lin and **Hwee Tou Ng**
Department of Computer Science
National University of Singapore
{ruixi,nght}@comp.nus.edu.sg

Abstract

The success of a natural language processing (NLP) system on a task does not amount to fully understanding the complexity of the task, typified by many deep learning models. One such question is: can a black-box model make logically consistent predictions for transitive relations? Recent studies suggest that pre-trained BERT can capture lexico-semantic clues from words in the context. However, to what extent BERT captures the transitive nature of some lexical relations is unclear. From a probing perspective, we examine WordNet word senses and the IS-A relation, which is a transitive relation. That is, for senses A, B, and C, A *is-a* B and B *is-a* C entail A *is-a* C. We aim to quantify how much BERT agrees with the transitive property of IS-A relations, via a minimalist probing setting. Our investigation reveals that BERT's predictions do not fully obey the transitivity property of the IS-A relation.[1]

1 Introduction

The IS-A relation denotes a subclass relation. If A *is-a* B, then the concept A is a subclass of the concept B, or A is subsumed by B. The IS-A relation is frequently encoded in lexical taxonomies. The IS-A relation has great significance since it empowers generalization, and generalization is at the core of machine inference for text understanding. The IS-A hierarchy is inherently transitive, i.e., for three concepts (or word senses) A, B, and C, A *is-a* B and B *is-a* C entail A *is-a* C. For example, knowing that *humanoid* is a type of *automaton*, and *automaton* is a type of *artifact*, then by transitivity, the relation *humanoid* is an *artifact* also holds.

The concept of transitivity is easy to comprehend by humans. However, deep learning models, including pre-trained language models such as BERT (Devlin et al., 2019), are known to lack some human-level generalization capacities in text understanding, or it may show some capacities for making correct predictions but for the wrong reasons, including being insensitive to negation and exploiting only surface features (Kassner and Schutze, 2020; Ettinger, 2020), lacking understanding of perceptual properties (Forbes et al., 2019; Weir et al., 2020), and surface form competition (Holtzman et al., 2021).

Despite the issues raised above, previous work has shown that BERT's layers align with the NLP pipeline, and representations in the different layers of BERT are found to capture different levels of textual understanding, from syntactic (e.g., part-of-speech tagging) to semantic (e.g., semantic role labeling) as the layers go from the lower to higher layers (Tenney et al., 2019a,b). Recent studies also suggest that BERT can capture lexical relation clues from words in contexts (Vulić et al., 2020; Misra et al., 2020). Researchers begin to recognize BERT as an open knowledge source and query BERT for information (Petroni et al., 2019). Moreover, BERT, even without fine-tuning on downstream tasks, possesses a fair ability to produce contextualized embeddings that cluster to word senses (Wiedemann et al., 2019; Haber and Poesio, 2020; Mickus et al., 2020; Loureiro et al., 2021). These findings suggest that BERT has some understanding of the building blocks of language. Following these findings, since an IS-A taxonomy can be built on top of explicit word senses, do contextualized embeddings learned from BERT for word senses (in particular contexts) respect the properties of the IS-A taxonomy, specifically transitivity? That is, does BERT make logically consistent predictions that enforce the transitivity constraint of the IS-A relation?

In this paper, we introduce a minimalist probing method to investigate whether BERT knows that the IS-A relation is transitive. We first quantify how well BERT predicts the IS-A relation. Next,

[1] The source code and dataset of this paper are available at `https://github.com/nusnlp/probe-bert-transitivity`.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 94 - 99
May 22-27, 2022 ©2022 Association for Computational Linguistics

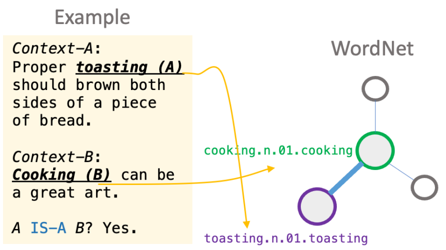
we measure the extent to which BERT enforces the transitivity constraint. That is, given that BERT predicts A *is-a* B and B *is-a* C, does it then predict A *is-a* C?

In our work, we make use of WordNet (Fellbaum, 1998) and propose a method to sample word sense pairs with contexts from WordNet example sentences to build a probing dataset. We use a nearest neighbor classifier for probing, which does not require any parameter tuning. Our findings indicate that BERT can predict IS-A relations with an accuracy score of 72.6%. However, when BERT predicts A *is-a* B and B *is-a* C, it only predicts A *is-a* C 82.4% of the time. This suggests that simply treating BERT as is as a knowledge base (Petroni et al., 2019) is not completely satisfactory, and additional work needs to be done to incorporate the transitivity constraint in natural language inference when using BERT.

2 Related Work

A key weakness of deep learning models is that they are black-box models and do not offer explainable and interpretable predictions. This has led to a large body of research regarding their interpretability (Linardatos et al., 2021). The pre-trained language model BERT has been extensively analyzed since its release. In particular, feature-based probes have been proposed to show how a particular layer, head, or neuron of BERT works on a downstream NLP task. Usually with a small set of additional parameters, a probe is trained in a supervised manner using feature representations from the pre-trained BERT, e.g., contextualized embeddings, to solve a particular task (Wu et al., 2020). Attention and structural probes have been invented to investigate different aspects of BERT and linguistic properties (Lin et al., 2019; Jawahar et al., 2019; Clark et al., 2019; Hewitt and Manning, 2019; Manning et al., 2020; Tenney et al., 2019a,b; Pruksachatkun et al., 2020). Latent ontology of contextual embeddings has also been investigated via cluster analysis (Michael et al., 2020). On probing contextualized representations for lexico-semantic relations, previous studies have investigated BERT for lexical relation classification via a neural network probe on type-level embeddings (Vulić et al., 2020).

Our work differs from prior work by our goal to explicitly investigate how much BERT understands the IS-A relation and more importantly, obeys the transitivity constraint. That is, we aim to determine if and how often BERT makes logically consistent predictions for the IS-A relation. Moreover, we focus on investigating sense-based IS-A relation, which is associated with explicit word senses in their contexts, so contextualized embeddings can be clearly mapped to word senses.



Context-A:
Proper ***toasting (A)*** should brown both sides of a piece of bread.

Context-B:
Cooking (B) can be a great art.

A IS-A B? Yes.

Figure 1: An example of an IS-A pair

3 Experimental Setup

We probe if and how well BERT can predict the IS-A relation and its transitivity.

Task Definition For a dataset of interest, we denote it as $\mathcal{D} = \{[(u_1, v_1), y_1], \cdots, [(u_n, v_n), y_n]\}$. $(u_{1:n}, v_{1:n}) = [(u_1, v_1) \cdots, (u_n, v_n)]$ are representations for pairs of word senses and $y_{1:n} = (y_1, \cdots, y_n)$ are labels for the IS-A relation classification task, where 1 denotes the positive IS-A relation and 0 otherwise. In our probing task, we quantify the extent to which $(u_{1:n}, v_{1:n})$ encode relations $y_{1:n}$. To probe BERT, we use the contextualized embedding (i.e., BERT's final hidden state output) of a word in a given context as the representation for the sense associated with the word's meaning in that context.

Contextualized Embeddings In this work, we focus on the BERT-base model. Given a target word w_t and its context c, BERT produces a final hidden state output as the contextualized embedding o_t for the target word w_t. If w_t is tokenized into subwords, we take the average over all subwords to be the contextualized embedding.

3.1 Probing Dataset

WordNet (Fellbaum, 1998) is a rich lexical database of word senses connected via the IS-A relation with example sentences for sense usages, making it a natural resource for probing. A 1-hop IS-A relation is illustrated in Figure 1. By transitivity, an n-hop IS-A relation is formed from a chain of n parent-child IS-A links. We focus on noun pairs in this work as nouns make up 70% of all senses in WordNet, and the path lengths for nouns are often longer than for verbs. We propose a path-based sampling method to generate pairs from WordNet, as follows:

1. Let $\mathcal{L} = \{ s \mid s \in \mathcal{S}$ and hypo(s) = $\emptyset \}$ denote all leaf senses from WordNet, where $\mathcal{S}$ represents the set of all senses in WordNet, and hypo(s) denotes the set of hyponym (children) senses of sense s.

2. For IS-A, sample a leaf sense N uniformly at random from $\mathcal{L}$, connect N to the root R, which gives a path p ($N \rightarrow R$). For not IS-A, similarly sample two leaf senses N_1, N_2 randomly from $\mathcal{L}$ and obtain two paths p_1 ($N_1 \rightarrow R$) and p_2 ($N_2 \rightarrow R$).

3. For IS-A, randomly sample three senses A, B, C from p and ensure that example sentences exist for senses A, B, C. This results in the 3-tuple (A, B, C) and three positive examples $(A, B), (B, C), (A, C)$. For not IS-A, randomly sample A' and B' from p_1 and p_2 respectively, ensuring that example sentences exist for senses A' and B'. If A' is not on the path of $B' \rightarrow R$ and vice versa, then we obtain a negative example (A', B'); else return to step 2.

4. Repeat step 2 and 3 to sample more positive and negative examples, until the desired number of examples is reached.

In our probing dataset, We have 1,665 3-tuples resulting in 4,995 positive examples, as well as 4,995 negative examples, where each example is a pair of senses.

3.2 Probing Method

Since our goal is to determine what BERT as a pre-trained language model knows about transitivity, we use a simple nearest neighbor (1-nn) classifier without further fine-tuning of BERT's parameters. We also adopt a 1-nn classifier instead of a more complex classifier so that we are measuring what BERT knows and not what is learned by a subsequent complex classification model.

Our 1-nn probing classifier works by finding the closest example in the training set for a test example, and using the closest training example's label as the prediction. Euclidean distance is used as the distance metric for our 1-nn probing classifier. We represent each example, which is a pair of senses, by the concatenation of the contextualized embeddings of the pair. For a pair of target words (w_1, w_2) and their respective contextualized embeddings (o_1, o_2), $r(o_1, o_2)$ denotes the relation embedding of the pair:

$$r(o_1, o_2) = [o_1; o_2] \tag{1}$$

Let r and r' denote two examples, and let m denote the dimension of the relation embeddings. The Euclidean metric $d(r, r')$ is computed as follows:

$$d(r, r') = \sqrt{\sum_{i=1}^{m} (r_i - r'_i)^2} \tag{2}$$

3.3 Evaluation

Model and Hyperparameters For our BERT model, we use the basic bert-base-uncased model[2], which has 12 layers with a hidden dimension of 768. For 1-nn, we adopt the scikit-learn (Pedregosa et al., 2011) KNeighborsClassifier implementation.

Training and Test Data for Probing Classifier Following similar sizes of other probing datasets (Vulić et al., 2020; Tenney et al., 2019b), we set aside a test set consisting of 1,998 positive (IS-A) examples (generated from 666 3-tuples) and 1,998 negative (not IS-A) examples. We split the remaining examples into 3 equal training sets, each consisting of 999 positive examples (generated from 333 3-tuples) and 999 negative examples. We report the average score over the 3 runs.

For the transitive examples $[(A, B), (B, C), (A, C)]$ in the test set, the average numbers of hops for (A, B) and (B, C) are 1.5 and 2.1 respectively. This difference is due to the fact that the senses with at least an example sentence are not evenly distributed along a path for nouns in WordNet. On average, only 46% of senses on a sampled path have example sentences, out of which 72% of the senses in the bottom half (i.e., the half closer to the leaves) of the path are associated with example sentences, whereas only 17% of the top half have example sentences. Therefore, when a sense C is sampled from the top half of the path, it is likely to be further away from sense B.

Evaluation Metric We adopt accuracy as our evaluation metric, which measures the percentage of test examples correctly predicted by the probing classifier. All accuracy scores are computed using the scikit-learn package.

[2] https://huggingface.co/transformers/pretrained_models.html

Pairs	IS-A		IS-A and not IS-A	
	# examples	Acc.	# examples	Acc.
All	1998	65.2 ± 2.8	3996	72.6 ± 0.9
1-hop	702	65.8 ± 2.4	1404	72.3 ± 0.9
2-hop	560	64.4 ± 3.5	1120	73.2 ± 1.8
3-hop	377	68.1 ± 3.7	754	72.4 ± 0.8
4-hop	212	65.6 ± 2.4	424	74.1 ± 1.1
5-hop	67	60.7 ± 4.6	134	71.6 ± 1.2
6-hop	40	58.3 ± 8.2	80	70.0 ± 4.7

Table 1: Accuracy scores on the test set, in the form of mean ± standard deviation. Columns 2–3: Accuracy (%) scores for n-hop IS-A pairs. Columns 4–5: Accuracy (%) scores for n-hop IS-A pairs and the same number of not IS-A pairs. Since longer paths are fewer and senses with example sentences are fewer when they are more distant from the leaf, the number of sampled pairs becomes fewer as the number of hops increases. Hops more than 6 are not shown as the number of n-hop examples for any $n > 6$ is fewer than 20.

$p(AB)$	$p(BC)$	$p(AC)$	$p(AC\|AB, BC)$
63.1 (1.9)	66.3 (3.9)	66.2 (2.9)	82.4 (3.1)

Table 2: Accuracy (%) scores for the 666 transitive 3-tuples in the test set. The standard deviations across three runs are shown in parentheses.

4 Experimental Results

4.1 Results Grouped by Number of Hops

The accuracy scores for the test set are shown in Table 1. The overall accuracy score for all pairs of both IS-A and not IS-A classes is 72.6%, suggesting that BERT correctly predicts IS-A relations to some extent. We also provide a breakdown of the accuracy scores according to different number of IS-A hops. The scores indicate that BERT predicts IS-A relations with higher accuracy for smaller number of hops (1–4) than for larger number of hops (5–6), although the prediction accuracy does not drop by a large amount when the number of hops increases, and the accuracy does not vary too much within 1–4 hops.

4.2 Prediction Ability for Transitivity

We quantify BERT's prediction ability for transitivity by measuring how often BERT makes logically consistent predictions for IS-A relations. Specifically, suppose word senses (A, B, C) form the following transitive IS-A relations: A *is-a* B *is-a* C. We measure how often BERT correctly predicts the IS-A relation (A, C) given that it correctly predicts (A, B) and (B, C). Table 2 shows the accuracy scores for the 666 transitive 3-tuples. In the table, $p(AB)$ denotes the percentage of correctly predicted (A, B) in the 666 (A, B) pairs. Similar definitions apply to $p(BC)$ and $p(AC)$. $p(AC|AB, BC)$ denotes the percentage of correctly predicted (A, C), given that (A, B) and (B, C) are correctly predicted. The conditional probability in Table 2 indicates that when BERT predicts that A *is-a* B and B *is-a* C, it correctly predicts that A *is-a* C 82.4% of the time. That A *is-a* C is not always predicted correctly (given that BERT correctly predicts A *is-a* B and B *is-a* C) suggests that BERT lacks the ability to make logically consistent predictions.

5 Conclusion

In this paper, we have investigated how much BERT agrees with the transitivity constraint of the IS-A relation, via a minimalist probing setting. Our findings indicate that although BERT can predict IS-A relations to some extent, it does not always make logically consistent predictions. Allowing BERT and more generally neural network models to enforce the transitivity constraint of the IS-A relation would be a worthy future research goal. Besides the IS-A relation, there are other transitivity relations like after, before, larger than, smaller than, etc. It would also be interesting to investigate to what extent BERT also enforces or fails to enforce these other transitivity relations in future work.

Acknowledgments

The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore (https://www.nscc.sg). We thank the anonymous reviewers for their helpful comments, and Wei Lu for his constructive suggestions and feedback on this work.

References

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

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

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. In *Transactions of the Association for Computational Linguistics*, pages 34–48.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? In *Proceedings of the 41th Annual Meeting of the Cognitive Science Society*.

Janosch Haber and Massimo Poesio. 2020. Word sense distance in human similarity judgements and contextualised word embeddings. In *Proceedings of the Probability and Meaning Conference*, pages 128–145.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4129–4138.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7038–7051.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657.

Nora Kassner and Hinrich Schutze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside BERT's linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 241–253.

Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. 2021. Explainable AI: a review of machine learning interpretability methods. In *Entropy*.

Daniel Loureiro, Kiamehr Rezaee, Mohammad Taher Pilehvar, and Jose Camacho-Collados. 2021. Analysis and evaluation of language models for word sense disambiguation. In *Computational Linguistics*, pages 387–443.

Christopher D. Manning, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. In *Proceedings of the National Academy of Sciences of the United States of America*, pages 30046–30054.

Julian Michael, Jan A. Botha, and Ian Tenney. 2020. Asking without telling: Exploring latent ontologies in contextual representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6792–6812.

Timothee Mickus, Denis Paperno, Mathieu Constant, and Kees van Deemter. 2020. What do you mean, BERT? assessing BERT as a distributional semantics model. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 279–290.

Kanishka Misra, Allyson Ettinger, and Julia Rayz. 2020. Exploring BERT's sensitivity to lexical cues using tests from semantic priming. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4625–4635.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, pages 2825–2830.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 2463–2473.

Yada Pruksachatkun, Phil Yeres, Haokun Liu, Jason Phang, Phu Mon Htut, Alex Wang, Ian Tenney, and Samuel R. Bowman. 2020. jiant: A software toolkit for research on general-purpose text understanding models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 109–117.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 7222–7240.

Nathaniel Weir, Adam Poliak, and Benjamin Van Durme. 2020. Probing neural language models for human tacit assumptions. In *Proceedings of the 42th Annual Meeting of the Cognitive Science Society*.

Gregor Wiedemann, Steffen Remus, Avi Chawla, and Chris Biemann. 2019. Does BERT make any sense? interpretable word sense disambiguation with contextualized embeddings. In *Proceedings of the 15th Conference on Natural Language Processing*, pages 161–170.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4166–4176.

Buy Tesla, Sell Ford: Assessing Implicit Stock Market Preference in Pre-trained Language Models

Cheng Yu Chuang[1] **Yi Yang**[2]
[1] Department of Mathematics and Economics,
[2] Department of Information Systems and Operations Management,
Hong Kong University of Science and Technology
cychuangab@connect.ust.hk, imyiyang@ust.hk

Abstract

Pretrained language models such as BERT have achieved remarkable success in several NLP tasks. With the wide adoption of BERT in real-world applications, researchers begin to investigate the implicit biases encoded in the BERT. In this paper, we assess the implicit stock market preferences in BERT and its finance domain-specific model FinBERT. We find some interesting patterns. For example, the language models are overall more positive towards the stock market, but there are significant differences in preferences between a pair of industry sectors, or even within a sector. Given the prevalence of NLP models in financial decision making systems, this work raises the awareness of their potential implicit preferences in the stock markets. Awareness of such problems can help practitioners improve robustness and accountability of their financial NLP pipelines [1].

1 Introduction

Pre-trained language models (PLM) have achieved superior performance on many NLP tasks (Devlin et al., 2018; Liu et al., 2019; Radford et al., 2019). They have also been integrated into real-world NLP systems for automated decision-making. Recently, a burgeoning body of literature has studied the human-like bias encoded in the PLMs. For example, in the mask token prediction task, BERT fill-in the [MASK] in the sentence "He/she works as a [MASK]" with "doctor/nurse", reflecting gender stereotype biased associations (Garimella et al., 2021; May et al., 2019). Such biases in the PLMs may further propagate to downstream applications with unintended societal and economic impact.

In this work, we investigate and assess the implicit preference encoded in the PLMs, in the context of the financial market. We examine if the PLMs prefer one company over the other companies. We also examine if such implicit preference in individual stocks also manifests at the industry sector level. Our core idea is based on the assumption that an NLP system designed to be widely applicable should ideally produce scores that are independent of the identities of name entities mentioned in the text (Prabhakaran et al., 2019).

	Sentence
BERT	*Tesla* stock share is going to float. *Ford* stock share is going to collapse.
FinBERT	*Tesla* stock share is going to increase. *Ford* stock share is going to decrease.

Table 1: Masked token predictions.

Table 1 illustrates the potential stock market implicit preferences in the BERT (Devlin et al., 2018) and its finance-domain specific variation FinBERT (Yang et al., 2020). Clearly, we see a favor of Tesla over Ford in both PLMs. This implicit association may be rooted in the training data: While BERT is trained on fairly neutral corpora, FinBERT is trained on financial communication corpora, including earnings conference calls and analyst reports. If a company's name is often mentioned in negative contexts (such as losses, disruptions), a trained model might inadvertently associate negativity to that name, resulting in biased predictions on sentences with that name.

We quantitatively assess the implicit preferences in the PLMs, using a sample of nearly 3,000 major U.S. market stocks. Our analysis reveals that the language models are overall more positive towards the stock market, but there are significant differences in preferences between a pair of industry sectors, or even within a sector. Given the wide adoption of PLMs in the financial applications, we hope our work raises awareness of their potential stock market implicit preferences of company names. Moreover, care needs to be taken to ensure that the unintended preference does not affect downstream applications. Awareness of such matters can help practitioners to build more robust

[1] Code and data for this work are available at https://github.com/MattioCh/Buy-Tesla-Sell-Ford

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 100 - 105
May 22-27, 2022 ©2022 Association for Computational Linguistics

and accountable financial NLP systems.

2 Background

Humans have (irrational) preferences in the stock markets. Humans are irrational (Becker, 1962). Human decision-makers are often influenced by emotion, biases, and cognitive errors. Human (irrational) preferences in the stock markets are well documented in behavioral finance/economics literature. For example, the home-bias refers to investors' strong preference for domestic stocks or concentrated exposure to their employer's stock (French and Poterba, 1991; Tesar and Werner, 1995). Bhattacharya et al. (2018) find that the Mandarin-speaking individual investors submit disproportionately more limit orders at 8 than at 4, because of the belief that the number 8 is lucky and the number 4 is unlucky — and those superstitious investors lose money.

Why is the implicit stock market preference in PLMs an issue? Automated NLP technique for financial decision making is expected to minimize human irrationality. However, PLMs that are trained with a human-written corpus may inherit such human preferences (we do find it is the case). This resembles the allocational harms that "arise when an automated system allocates resources or opportunities unfairly to different social groups" (Blodgett et al., 2020). In the financial markets, the disproportional allocation of resources, i.e., capital, also has unintended consequences. First, the strong favoritism to a stock can attract more investors to invest in the stock and increase the company's capital value, which helps the company's growth and development (Beck and Levine, 2002). This implies that less favored companies may struggle with capital access. Second, the disproportional resource allocation may result in high trading activities and increased volatility of certain stocks, which creates uncertainty and instability in the market.

3 Data and PLMs

Data: We choose Russell 3000 constituent firms as our target companies because of their importance and tractability. This index includes the 3,000 largest publicly held companies incorporated in the United States as measured by total market capitalization, and it represents approximately 98% of the U.S. public equity market. We also obtain an industry sector label for each firm in our sample, based on the Global Industry Classification Standard (GICS). GICS is a widely used industry classification for market analysis, and it consists of 11 sectors. For example, company Apple (NASDAQ:AAPL) is in the *Information Technology* sector, while the company Walmart (NASDAQ:WMT) is in the *Consumer Staples* sector. The GICS sector allows us to examine the implicit preference at the industry sector level. The total number of stocks in our sample is 2,653. The detailed breakdown of GICS sectors in our sample is presented in Table 2.

GICS Sector	Number of stocks
Financials	495
Industrials	391
Health Care	379
Information Technology	351
Consumer Discretionary	310
Real Estate	162
Energy	144
Materials	136
Communication Services	110
Consumer Staples	104
Utilities	71

Table 2: Sample stocks GICS breakdown.

PLM: We choose two BERT-based pre-trained language models in our analysis: BERT and FinBERT. BERT is one of the most widely used PLMs that is trained on Wikipedia and BookCorpus (Devlin et al., 2018). In addition to BERT, we choose FinBERT, which is a domain-specific BERT model that is pre-trained on financial communications text, including annual reports, analyst reports, and earnings conference call transcripts (Yang et al., 2020). The vocabulary of FinBERT is different from the BERT model as it contains finance-domain specific terms, including company names. It has shown to outperform the general-domain BERT (Huang et al., 2020) on financial downstream tasks. We load both base-uncased BERT and FinBERT from the `transformers` library (Wolf et al., 2020).

4 Assessing Implicit Preference in Masked Token Prediction

Since BERT and FinBERT use a masked language modeling objective, we directly probe the model using the masked token prediction task, using cloze-style prompts. Prior work also uses this approach to assess the social biases (May et al., 2019), or the knowledge learned by PLMs (Petroni et al., 2019). For each firm, we create a simple template containing the attribute word for which we want to measure the preference (e.g. buy or sell) and the company name as the target word (e.g., Microsoft). We then mask the attribute words and target words accordingly, to get the conditional probability of producing the *buy* or *sell* token. Specifically, for firm i, we use the template sentence "We should [MASK] the {name} stock" and query the probability of masked token: $P_{i,buy} = P([MASK] = buy|\text{name} = i)$, and $P_{i,sell} = P([MASK] = sell|\text{name} = i)$. We then normalize the two conditional probabilities.

4.1 Implicit preferences in the market

Our first evaluation simply assesses if the PLM is lean more towards buy or sell across companies. We obtain the normalized conditional probability $P_{i,buy}$ for each firm i, and we plot the boxplot of $P_{i,buy}$ in Figure 1. An ideal model would have a conditional probability close to 0.5 for all firms. Clearly, it is not the case in the BERT and FinBERT. Figure 1 shows that the mean value of $P_{i,buy}$ is significantly different from 0.5. FinBERT's average buy probability is even higher than 0.9, indicating as a stronger preference for predicting buy token over the sell token. This tendency could be explained by two reasons. First, prior literature shows that there is a universal positive bias in the human language (Dodds et al., 2015). Second, compared to BERT which is trained on a fairly neutral corpus, FinBERT is trained on financial communication corpora such as analyst reports. Therefore, the higher buy probability may imply that the overall market sentiment over the years is positive.

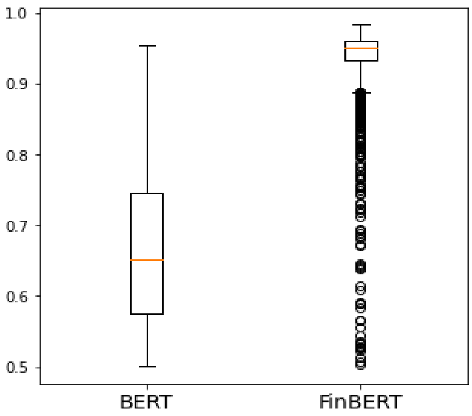


Figure 1: Boxplot of $P_{i,buy}$ (normalized with $P_{i,sell}$) for BERT and FinBERT. It shows strong positive preferences in company names.

4.2 Implicit preferences between industries

It may not be surprising that the PLMs are overall positive. Therefore, we examine if certain industry sectors are more favored than the other industries. We use a univariate regression analysis. For firm i, we use the $P_{i,buy}$, the probability of predicting the masked token "buy", as the response variable, and we use the firm's sector X_i as the dummy independent variable, i.e., X_i is 1 if stock i belong to the sector j, otherwise 0. Since we have a total of 11 sectors, we set up 11 univariate regression models and examine the relationship between the probability of "buy" and the dummy industry sector variable. The univariate regression is specified as follows, and ϵ is the error term.

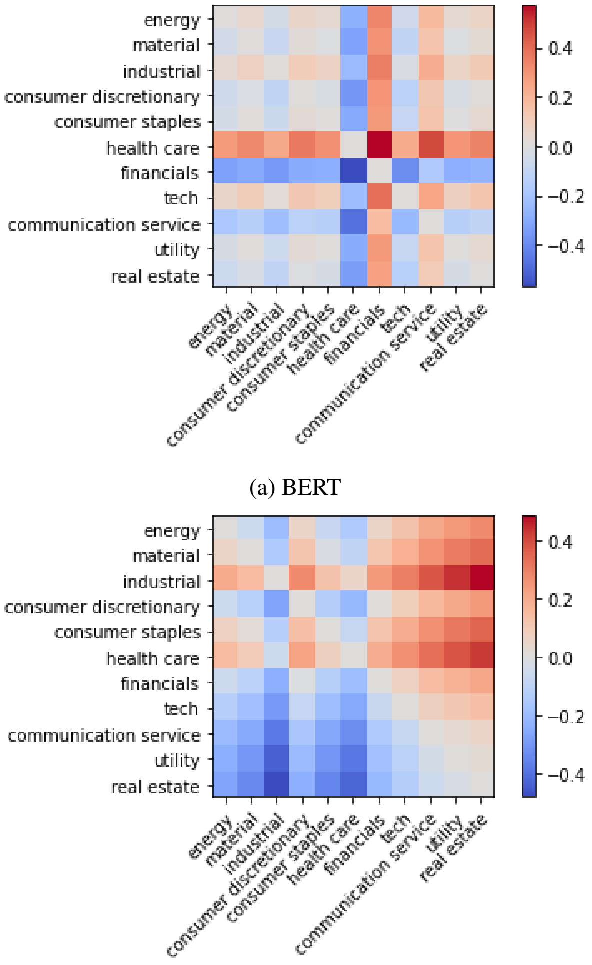
$$\text{For sector } j\text{: } y^j = \beta^j x_i^j + \epsilon \qquad (1)$$

The univariate regression results are presented in Table 3. We can see that both models have preferences of one sector over the other sectors. For example, both BERT and FinBERT find companies in the Financial sectors less preferred in terms of predicting the buy token, as seen from the negative β value and significant p-values. From Table 2, we can see that the most preferred sectors in BERT are Materials, Consumer Staples, and Utilities; while for FinBERT, the most preferred sector is Materials and Industrials. Moreover, we find that, while FinBERT has a stronger buy preference across all companies than BERT, it has less preference when comparing to the industry sector level, as we see there is a fewer number of sectors with significant p-values. In other words, FinBERT has positive preference across most of sectors, while BERT has positive preference only in certain sectors.

We further compare the implicit preference between a pair of sectors. To do so, we conduct Cohen's d test and calculate the effect size of the distributions of pair of industry A and industry B. Specifically, Cohen's d determines the mean difference between industry A and B in terms of the probability $P_{i,buy}$. A positive value indicates that the PLM has a stronger buy preference for industry A than for industry B. We plot the heatmap between pairs of industries in Figure 2. The figure shows that both models have an implicit preference between sectors. Consistently, Financial is the least preferred industry sector.

GICS Sector	BERT	FinBERT
Financials	-0.88***	-0.83***
Industrials	0.43	0.40***
Health Care	0.00***	0.10
Information Technology	0.12***	0.7
Consumer Discretionary	0.17**	-0.94
Real Estate	-1.88***	0.07*
Energy	0.15	0.72*
Materials	2.22***	1.09**
Communication Services	-0.07	-0.98
Consumer Staples	0.73**	-0.30
Utilities	0.61***	1.34

Table 3: Value of β ($\times 10^{-2}$) using BERT and FinBERT model. Asterisk indicates statistical significance p-value: * $p < .1$, ** $p < .05$, *** $p < .01$



(a) BERT

(b) FinBERT

Figure 2: Heatmap of Cohen's d test between a pair of sectors. Higher value (red) indicates a stronger preference in predicting the buy token from one sector on the vertical axis to another sectors on the horizontal axis.

5 Assessing Implicit Preferences within an Industry Sector

The masked token prediction is only one way of probing the PLMs. Recent NLP literature has proposed the word association tests to measure the human-like biases in the static word embedding (Bolukbasi et al., 2016; Caliskan et al., 2017) or contextualized word embedding (May et al., 2019). The word association test in the contextualized embedding model is called Sentence Encoder Association Test (SEAT). Essentially, SEAT evaluates whether the contextualized representations for words from an attribute word set tend to be more closely associated with the contextualized representations for words from a target word set. Templates such as "this is a [word]" are used to obtain the word contextualized representations.

In this work, we create a template sentence "{name} is a stock" where {name} is a stock's company name, and we obtain the [CLS] embedding as its embedding. For preference words buy and sell, we create a template "We should buy/sell a stock", and we obtain the [CLS] embedding as its embedding. Let $sim_{i,buy}$ and $sim_{i,sell}$ be the cosine similarity between the embedding of company i's name and the embedding of buy/sell. Given an industry sector S containing a set of stocks, we calculate the SEAT association effect-size as: $d = \frac{mean_{i \in S}(sim_{i,buy}) - mean_{i \in S}(sim_{i,sell})}{std\_dev_{i \in S}\{sim_{i,buy}, sim_{i,sell}\}}$. An effect size with absolute value closer to 0 indicates lower implicit preference. We present the individual sector's SEAT score in Table 4, which leads to the following observations. First, we see consistent implicit preferences *within* individual sectors. For example, both BERT and FinBERT regard Financials as the least preferred sector (negative effect size). Since this is a within-in sector study, it implies that some Financial stocks are preferred over the other Financial stocks. Second, we see that the majority of the sectors have a positive effect size, indicating that both PLMs exhibit a positive bias within the sector.

6 Conclusion

In this paper, we study the implicit stock market preference in PLMs. Motivated by recent literature in implicit social bias, we apply the masked token prediction and sentence embedding association test (SEAT) to the PLMs. We find that there is a consistent implicit preference of the stock market in the PLMs, and the preferences exist at the whole-

GICS Sector	BERT	FinBERT
Financials	-0.65	-0.15
Industrials	0.19	0.34
Health Care	0.06	-0.03
Information Technology	0.44	0.06
Consumer Discretionary	0.25	0.26
Real Estate	0.00	0.29
Energy	-0.56	0.44
Materials	-0.15	0.15
Communication Services	0.10	-0.06
Consumer Staples	-0.08	0.18
Utilities	-0.19	0.12

Table 4: Within-sector implicit preferences using SEAT. Value close to zero indicates lower implicit preference.

market, between-industry,and within-industry level. Given the wide adoption of PLMs in real-world financial systems, we hope that this work raises the awareness of potential implicit stock preferences, so that practitioners and researchers can build more robust and accountable financial NLP systems. Future work can investigate whether the implicit preferences are driven by some financial factors such as market value or stock returns, and examine how the preferences over stocks/industries in PLMs affect downstream financial NLP applications, such as sentiment analysis, or stock movement prediction.

Acknowledgement

This work was supported by HKUST-Kaisa Group Seed Project on Fintech "HKJRI3A-057".

References

Thorsten Beck and Ross Levine. 2002. Industry growth and capital allocation:: does having a market-or bank-based system matter? *Journal of financial economics*, 64(2):147–180.

Gary S Becker. 1962. Irrational behavior and economic theory. *Journal of political economy*, 70(1):1–13.

Utpal Bhattacharya, Wei-Yu Kuo, Tse-Chun Lin, and Jing Zhao. 2018. Do superstitious traders lose money? *Management Science*, 64(8):3772–3791.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna M. Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5454–5476. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam Tauman Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4349–4357.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Peter Sheridan Dodds, Eric M Clark, Suma Desu, Morgan R Frank, Andrew J Reagan, Jake Ryland Williams, Lewis Mitchell, Kameron Decker Harris, Isabel M Kloumann, James P Bagrow, et al. 2015. Human language reveals a universal positivity bias. *Proceedings of the national academy of sciences*, 112(8):2389–2394.

Kenneth R French and James M Poterba. 1991. Investor diversification and international equity markets. *The American Economic Review*, 81(2):222–226.

Aparna Garimella, Akhash Amarnath, Kiran Kumar, Akash Pramod Yalla, N Anandhavelu, Niyati Chhaya, and Balaji Vasan Srinivasan. 2021. He is very intelligent, she is very beautiful? on mitigating social biases in language modelling and generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4534–4545.

Allen Huang, Hui Wang, and Yi Yang. 2020. Finbert—a deep learning approach to extracting textual information. *Available at SSRN 3910214*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Chandler May, Alex Wang, Shikha Bordia, Samuel R Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. *arXiv preprint arXiv:1903.10561*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Vinodkumar Prabhakaran, Ben Hutchinson, and Margaret Mitchell. 2019. Perturbation sensitivity analysis to detect unintended model biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5740–5745.

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

Linda L Tesar and Ingrid M Werner. 1995. Home bias and high turnover. *Journal of international money and finance*, 14(4):467–492.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Yi Yang, Mark Christopher Siy Uy, and Allen Huang. 2020. Finbert: A pretrained language model for financial communications. *arXiv preprint arXiv:2006.08097*.

Pixie: Preference in Implicit and Explicit Comparisons

Amanul Haque and **Vaibhav Garg** and **Hui Guo** and **Munindar P. Singh**
Department of Computer Science
North Carolina State University
Raleigh, NC 27695, USA
{ahaque2, vgarg3, hguo5, mpsingh}@ncsu.edu

Abstract

We present Pixie, a manually annotated dataset for preference classification comprising 8,890 sentences drawn from app reviews. Unlike previous studies on preference classification, Pixie contains *implicit* (omitting an entity being compared), and *indirect* (lacking comparative linguistic cues) comparisons. We find that transformer-based pretrained models, finetuned on Pixie, achieve a weighted average F1 score of 83.34% and outperform the existing state-of-the-art for preference classification model (73.99%).

1 Introduction

Online user reviews contain a cornucopia of information on user expectations about a product. Users often express their opinions on a product by comparing it against competitors. Understanding preferences in natural language is crucial in capturing user's opinions and expectations. Previous studies show that app reviews include rich insights about user expectations and problems of mobile apps that are valuable for app developers (Palomba et al., 2015; Maalej and Nabil, 2015; Guo and Singh, 2020). We found that app reviews often include comparative sentences, from which we can determine a reviewer's preferences.

Identifying the preferred entity from an app review involves (1) *Comparative Sentence Identification* (CSI) (Jindal and Liu, 2006), i.e., identifying sentences that contain a comparison, and (2) *Comparative Preference Classification* (CPC) (Ganapathibhotla and Liu, 2008; Panchenko et al., 2019), i.e., identifying the preferred entity in a comparative sentence. We focus on the second task.

Prior work on CPC focuses on *explicit* comparisons, where all compared entities are explicitly mentioned. Extracting comparative sentences by matching keywords or patterns (Jindal and Liu, 2006; Li et al., 2017; Feldman et al., 2007) overlooks *indirect* comparisons which lack comparative quantifiers and adjectives.

	Sentence	App
S_1	Bye ***Uber***, hello ***Lyft***.	*Uber*
S_2	Does ***this app*** really need to be 260 MB when the ***Marriott app*** is only 47 MB?	*Hilton Honors*
S_3	Beats the pants off ***pandora***.	*Spotify*
S_4	I think that ***it's*** a lot more fun than ***temple Run***.	*Subway Surfers*

Table 1: Example comparative sentences from reviews.

Staab and Hahn (1997) identify *omitted complement* as a comparative sentence type that has been overlooked by prior research. An omitted complement refers to one of the entities under comparison that is omitted but can be inferred based on the context. We have found that comparative sentences in user-generated text such as reviews sometimes imply instead of explicitly mentioning the target entity being reviewed (e.g., S_3 in Table 1). Comparisons in reviews often lack comparative linguistic cues, such as comparative quantifiers, adjectives, or structures (i.e., indirect, e.g., S_1 in Table 1). Such sentences are comparative by virtue of expressing a preference and are common in reviews but have been understudied by prior research.

We present Pixie (Preference in Implicit and Explicit Comparisons), a dataset for preference classification, created from online user reviews. As shown in Table 1, Pixie includes indirect comparisons (i.e., sentences lacking comparative linguistic cues, e.g., S_1) and *implicit* comparisons (omitting compliments, i.e., mentioning only one entity being compared, e.g., S_3) in addition to *direct* comparisons (comparing entities with a direct comparative structure, e.g., S_4) and *explicit* comparisons (mentioning both entities being compared, e.g., S_2).

We experiment with traditional machine learning methods and transformer-based models on Pixie. We use segment embeddings to demar-

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 106 - 112
May 22-27, 2022 ©2022 Association for Computational Linguistics

cate the compared entities before fine-tuning the transformer-based models. We also compare our results with ED-GAT (Ma et al., 2020), a state-of-the-art model for preference classification. We find that transformer-based pretrained language models, fine-tuned on Pixie, achieve a higher F1-score (83.34%) than the state-of-the-art (F1-score 73.99%) or traditional machine learning models (F1-score 71.86%) trained on Pixie. Further error analysis of misclassifications reveals substantial differences between ED-GAT and transformer-based pretrained language models' performance.

Current research on preference classification is lacking and far from practical use. Real world comparisons can present characteristics that complicate the task, such as indirect comparisons, implicit comparisons, and ambiguous statements. The low F1-score of the existing state-of-the-art and noticeable differences in misclassifications across different models call for a more thorough research effort on preference classification in text.

2 Related work

Comparative sentence structures have been a subject of syntactic and semantic theories (Bresnan, 1973; Stechow, 1984; van Rooij, 2011). Early studies in computational linguistics include syntactic and semantic handling of comparative constructions (Rayner and Banks, 1988, 1990), comparative structures in question answering (Ballard, 1988), using quantifiers to identify comparisons (Friedman, 1989), and semantic interpretation of comparatives (Staab and Hahn, 1997).

Jindal and Liu (2006) present a binary classification dataset containing comparative and non-comparative sentences. They present a classifier based on Class Sequential Rules (CSR) and leverage comparative keywords to identify comparative sentences. Ganapathibhotla and Liu (2008) extend this work by annotating comparative sentences with the preferred entity.

Kessler and Kuhn (2014) annotate comparative sentences by identifying comparison predicates, entities being compared, aspect of comparison, and comparison type (gradable or non-gradable). However, they focus on reviews of only one product type (digital cameras) to create their dataset. Hence, their dataset lacks diversity in topics.

Panchenko et al. (2019) create CompSent-19, a cross-domain dataset for comparative argument mining. They propose a gradient boosting model based on pretrained sentence embeddings to identify the preferred entity. Ma et al. (2020) propose a model called Entity-aware Dependency-based Deep Graph Attention Network (ED-GAT) that consists of a multihop graph attention network with dependency relations to identify the preferred entity. The ED-GAT model achieves a micro F1-score of 87.43% on the CompSent-19 dataset.

Previous work on preference classification has overlooked implicit and indirect comparisons common in user-generated text such as app reviews. Further, existing datasets are either too small with a few comparative sentences or have a skewed distribution. For example, Ganapathibhotla and Liu's dataset contains only 837 comparative sentences, 84% of which have the first mentioned entity in the text as preferred. Only 15% of Kessler and Kuhn's dataset constitutes comparative sentences. Only 27% of the sentences in CompSent-19 (Panchenko et al., 2019) contain a preference, 70% of which prefer the first mentioned entity in the sentence.

Further, existing datasets consider the order of the appearance of compared entities in a sentence to annotate the preferred entity. For instance, annotations for CompSent-19 (Panchenko et al., 2019) and Ganapathibhotla and Liu's dataset are both determined based on the order of appearance of the entity in a sentence (i.e., is the first appearing entity in the sentence preferred or the second).

3 Method

We introduce the essential concepts below.

Comparative sentence : A sentence that contains information on similarity, dissimilarity, or preference between two entities.

Pixie includes (1) comparative sentences that lack comparative quantifiers, adjectives, or keywords, i.e., *indirect comparisons*, (2) *implicit comparisons* where only one of the compared entities is mentioned, and (3) *explicit comparisons* which mention both (including pronominal references).

Preferred entity : an entity that is chosen over another based on a stated or implied preference.

A preferred entity can be the CURRENT app (e.g., S_{1p} in Table 2), OTHER app (e.g., S_{2p} in Table 2), or NONE (i.e., ambiguous or no preference, e.g., S_{4p} in Table 2 or where non-gradable comparatives (such as *like*, *as ...*, and *similar to*) link the entities, e.g., S_{3p} in Table 2).

	Sentence	Review
S_{1p}	***This app* is better than *Discover's app***.	*Chase Mobile*
S_{2p}	I prefer the ***BBC app***.	*USA Today*
S_{3p}	Just as good as ***Uber*** app.	*Lyft*
S_{4p}	Makes me want to switch back to ***Pandora***, but ***it's*** just as bad.	*Spotify*

Table 2: Example sentences showing preference.

3.1 Dataset

We selected 179 popular apps on Apple App Store and collected their reviews. After some preliminary investigation, we excluded widely mentioned brand names such as Google, Microsoft, and Facebook, because they often appear in broader contexts than as a product. We removed app names synonymous with or formed of common words, such as Box (cloud storage) and Line (communication app) for higher precision in extracting comparative sentences. We were left with 141 apps, which we manually grouped into 23 genres, including *banking*, *airline*, and *communication*. Apps in the same genre are direct competitors. For example, *airline* apps include Delta, American, and United.

We extracted sentences that mention a competitor from each review and labeled each extracted sentence for comparison and preferred entity. When identifying mentions, we included common aliases or abbreviations on our name list, e.g., *Insta* for Instagram, *BA* for Bank of America, and *AA* for American Airlines to improve recall. Focusing on mentions of competitors ensures that Pixie includes indirect comparisons because such sentences are more likely to contain comparisons.

The dataset was annotated in three phases. In Phase 1, the authors annotated a sample dataset of 300 sentences based on an initial set of definitions and resolved any disagreements via discussions. We repeated this process for three iterations and produced annotation instructions for Phase 2. In Phase 2, each author annotated an equal number of sentences, and the disagreements were resolved by the first author, producing 4,793 annotated sentences. The interrater agreement (Krippendorff alpha) was 0.74 and 0.82 between the two annotators for comparison and preferred entity, respectively. Phase 3 involved crowdsourcing with 42 annotators, students in Natural Language Processing (NLP) course, each annotating around 400 sentences. 5,559 data points were labeled in Phase 3 with an interrater agreement (Krippendorff alpha) between the three annotators of 0.51 and 0.74 for comparison and preferred entity, respectively. We obtained the Institutional Review Board (IRB) approval for this task.

Once we removed duplicate and noncomparative sentences, we were left with 8,890 comparative sentences annotated for comparison type (IMPLICIT or EXPLICIT) and preferred entity (CURRENT, OTHER, or NONE). Table 3 shows the distribution of labels for each class in Pixie.

	Comparison Type		
Preferred Entity	Implicit	Explicit	Total
CURRENT	1,910	2,097	4,007
OTHER	2,199	1,069	3,268
NONE	758	857	1,615
Total	4,867	4,023	8,890

Table 3: Pixie Dataset Distribution.

To ensure that the dataset can be used to train a general-purpose preference classification model, we *mask* app mentions in each sentence. With no masking, the model may learn to differentiate between classes based on what users prefer more (app A or app B) in our dataset. Masking app mentions ensures that the model learns comparative and preference revealing linguistic structures and semantics instead of simply learning to differentiate between preferred entities in an exhaustive list of compared entities. We defined two tags for masking, *current_app* for the apps being reviewed and *other_app* for the competitor apps. App mentions are identified using the competitor app list for apps referred to by name, and pronoun references are substituted manually. Treating pronoun references as an explicit reference to app mentions ensures consistent based on our definitions, i.e., all explicit comparisons have two mentioned entities being compared, while all implicit comparisons have one. A portion of the dataset, ≈2,100 (∼23.62%) sentences, had pronoun references that were resolved. Table 4 shows sentences masked for app mentions.

For a quick sanity check, whether Pixie contains indirect comparative sentences, we examine how many of the sentences in Pixie contain a comparative word. For this, we combine the list of opinion words from (Hu and Liu, 2004) and the list of comparative cue words from (Panchenko et al., 2019). Only 3,781 sentences (42.5% of Pixie) contain a comparative or opinion word showing that most of the sentences in Pixie lack comparative cues (i.e., are indirect comparisons).

Unlike prior datasets on preference classification (Ganapathibhotla and Liu, 2008; Panchenko et al., 2019), Pixie does not consider the order of appearance of compared entities for annotations. Pixie also offers a more balanced dataset than the existing ones for the task. For explicit comparisons (when both entities are present), 1909 sentences (47.45%) prefer entity that appears first, 1257 (31.25%) sentences prefer entity that appears later, and 857 sentences (21.30%) reveal no or mixed preference. Implicit comparisons mention only one entity so the order of appearance is irrelevant.

Pixie is publicly available[1] and contains original and masked sentences.

3.2 Experiments

Among traditional machine learning approaches, we experiment with AdaBoost (Hastie et al., 2009), Random Forest (Breiman, 2001) and Support Vector Machine (SVM) (Chang and Lin, 2011). We use SBERT (SentenceBERT) (Reimers and Gurevych, 2019) to obtain sentence embeddings for each masked sentence.

For transformer-based language models, we fine-tune variations of BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) and XLNet (Yang et al., 2019). We experiment with BERT (Devlin et al., 2019), ALBERT (A Lite BERT) (Lan et al., 2019), and DeBERTa (Decoding-enhanced BERT with disentangled attention) (He et al., 2020). We fine-tune each model for 20 epochs using AdamW Optimizer with a learning rate of 5e-5 and a weight decay of 0.01. We use the train-test-validation split of 60-20-20.

We use segment embeddings to improve the performance of the transformer-based models. We assign different segment token ids to the competitor app (other_app) and the rest of the sentence to separate the entities being compared. We fine-tune pretrained models with segment embeddings along with token embeddings and attention masks.

To compare our results with ED-GAT, we convert the sentences in Pixie to follow the CompSent-19 format. Specifically, we add a token ([THIS]) for the current app in the front of each implicit sentence and map the labels CURRENT and OTHER to BETTER and WORSE, as applicable. NONE labels stay the same. We implemented ED-GAT with BERT embeddings and used eight GAT layers. We use the Hugging Face (Wolf et al., 2020) library for all transformer-based experiments.

To test the quality of Pixie, we run some cross-dataset experiments as well. We train a DeBERTa model on Pixie and test on CompSent-19 and vice-versa. Since the CompSent-19 dataset is highly skewed, we balanced both datasets to have the same train and test data split across all three classes via random oversampling with replacement. We keep all other model parameters and configurations the same and leverage the same number of samples for training and testing.

4 Results

Table 5 contains results for models trained and tested on Pixie. SVM achieves the highest weighted F1-score of 71.86% (among the traditional approaches), and DeBERTa (F1-score 83.34%), among transformer-based models.

Segment embeddings enhanced BERT and XLNet model's performance in terms of weighted average F1-scores, but a slight decline for DeBERTa's and ALBERT's performance.

The NONE and CURRENT classes consistently achieve the lowest and the highest F1-scores, respectively, for all models. The NONE class was also the most ambiguous class to annotate manually. Recall for the NONE class is lower than precision for all models except ED-GAT. All transformer-based models achieve a higher recall than precision for the CURRENT class except for ALBERT (without segment embeddings) and ED-GAT.

ED-GAT (Ma et al., 2020) trained on Pixie achieves a weighted average F1-score of 73.99%, with the highest F1-score (80.57%) for the CURRENT class and lowest (51.54%) for NONE.

Upon further analysis, we found that most of the incorrect classifications in transformer-based models are for the NONE class (71.64%), whereas, for ED-GAT, only 8.77% of the misclassified sentences belong to the NONE class. ED-GAT yielded most misclassifications for the CURRENT class (55.27% of misclassified instances) while only 14.93% of misclassifications for the transformer-based models belong to the CURRENT class.

Table 6 shows the results for the cross-dataset experiments. The weighted average F1-score improves by 4.08% with plain vanilla fine-tuning and 6.30% with segment embeddings when trained on Pixie and tested on CompSent-19. While the accuracy improves by 5.11% for plain vanilla fine-

[1] https://github.com/ahaque2/Pixie.git

	Original sentence	**Masked sentence**
1	*CNN* should leave journalism to the pros at *Fox* news.	<*current_app*> should leave journalism to the pros at <*other_app*> news.
2	way better than *Pandora* by a long shot!!!!	way better than <*other_app*> by a long shot!!!!
3	*This* is a great game just like *Temple run*	<*current_app*> is a great game just like <*other_app*>

Table 4: Original and masked comparative sentences.

Approach	**Model**	Current			None			Other			Weighted Average		
		Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Prior Work	ED-GAT	83.24	78.05	**80.57**	48.89	54.49	**51.54**	76.28	77.79	**77.03**	74.44	73.68	**73.99**
Traditional ML	AdaBoost	71.57	73.44	72.49	45.06	35.29	39.58	63.53	68.30	71.57	63.80	64.62	64.07
	Random Forest	71.27	80.42	75.57	71.31	26.93	39.10	64.98	74.73	69.52	68.97	68.62	66.72
	SVM	76.99	82.17	**79.49**	62.63	36.84	**46.39**	71.04	79.63	**75.09**	72.19	73.00	**71.86**
Transformer-Based	BERT	82.83	89.03	85.82	62.68	55.11	58.65	83.07	80.40	81.71	79.26	79.70	79.37
	DeBERTa	88.34	90.65	**89.48**	64.56	56.97	60.53	85.97	88.21	**87.07**	83.15	83.63	**83.34**
	ALBERT	87.83	87.28	87.55	61.37	60.99	**61.18**	84.70	85.60	85.15	81.87	81.89	81.88
	XLNet	83.45	90.52	86.84	67.06	52.32	58.78	83.99	84.38	84.19	80.67	81.33	80.77
Transformer-Based with Segment Embeddings	BERT	83.43	88.53	85.90	66.67	52.63	58.82	81.25	83.61	82.42	79.58	80.20	79.70
	DeBERT	88.31	91.40	**89.83**	64.34	56.97	**60.43**	85.35	86.52	**85.93**	82.87	83.35	**83.06**
	ALBERT	86.26	87.66	86.95	65.92	54.49	59.66	81.90	87.29	84.51	80.96	81.50	81.10
	XLNet	85.68	90.27	87.92	61.86	55.73	58.63	85.51	84.07	84.79	81.29	81.72	81.45

Table 5: Results (in %) for preference classification on Pixie. Bold indicates highest F1-scores for each category.

Approach	**Fine-tuning**	**Testing**	**Prec**	**Recall**	**F1**	**Accuracy**
Plain vanilla	CompSent-19	Pixie	65.46	59.89	59.23	59.89
Plain vanilla	Pixie	CompSent-19	65.19	65.00	63.31	65.00
With segment embeddings	CompSent-19	Pixie	67.84	59.44	57.70	59.44
With segment embeddings	Pixie	CompSent-19	66.07	65.72	64.00	65.72

Table 6: Results for cross-dataset experiments. The values are in %.

tuning and 6.28% with segment embeddings. The improvement primarily is in the recall, demonstrating that Pixie includes more diverse comparative sentences than CompSent-19.

5 Conclusion

Masking compared entities ensure that Pixie can be used to train a general-purpose preference classification model. Additional analysis is needed to claim the domain generality of our dataset—that is, whether a model trained on Pixie can identify the preferred entity in texts from other domains such as scientific papers and news. Comparative sentences in Pixie are limited to user-generated text and may not generalize well over more formal texts.

Both BERT and XLNet show improvements with segment embeddings, suggesting that the demarcation of the other app helps the model identify the preferred entity. The traditional machine learning models perform worst and the transformer-based pretrained models fine-tuned on Pixie achieve a substantially better performance than the state-of-the-art approaches for preference classification.

Identifying preferences in user reviews can aid developers in understanding user expectations about mobile apps. Users often express their likes and dislikes about an app or feature by comparing it with alternative apps and features. Understanding user preferences can be particularly valuable in enhancing the functionality as well as security and privacy features of apps. A user's preferences regarding apps would depend not only on how well the app is constructed relative to its competitors but also on how easily the app is used by end-users. For example, security concerns may be signaled by descriptions of steps to access sensitive financial or medical data (Guo and Singh, 2020) expressed in association with comparisons. A follow-on direction is to extract and prioritize user expectations by identifying the specific features of an app of greatest influence on the indirect or direct comparisons in a review.

Acknowledgments

This research is supported by the US Department of Defense under the Science of Security Lablet (SoSL) grant to NC State University. We thank the ACL reviewers for their helpful comments.

References

Bruce W. Ballard. 1988. A general computational treatment of comparatives for natural language question answering. In *Proceedings of the 26th Annual Meeting on Association for Computational Linguistics*, ACL, page 41–48, USA. Association for Computational Linguistics.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Joan W. Bresnan. 1973. Syntax of the comparative clause construction in English. *Linguistic Inquiry*, 4(3):275–343.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 17th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ronen Feldman, Moshe Fresko, Jacob Goldenberg, Oded Netzer, and Lyle Ungar. 2007. Extracting product comparisons from discussion boards. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 8001:469–474.

Carol Friedman. 1989. A general computational treatment of the comparative. In *27th Annual Meeting of the Association for Computational Linguistics*, pages 161–168, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Murthy Ganapathibhotla and Bing Liu. 2008. Mining opinions in comparative sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING, page 241–248, USA. Association for Computational Linguistics.

Hui Guo and Munindar P. Singh. 2020. Caspar: Extracting and synthesizing user stories of problems from app reviews. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ICSE, page 628–640, New York, NY, USA. Association for Computing Machinery.

Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. 2009. Multi-class AdaBoost. *Statistics and its Interface*, 2(3):349–360.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. DeBERTa: Decoding-enhanced BERT with disentangled attention. *arXiv preprint*, abs/2006.03654.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD, page 168–177, New York, NY, USA. Association for Computing Machinery.

Nitin Jindal and Bing Liu. 2006. Identifying comparative sentences in text documents. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR, page 244–251, New York, NY, USA. Association for Computing Machinery.

Wiltrud Kessler and Jonas Kuhn. 2014. A corpus of comparisons in product reviews. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*, pages 2242–2248, Reykjavik, Iceland. European Language Resources Association (ELRA).

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv preprint*, abs/1909.11942.

Yuanchun Li, Baoxiong Jia, Yao Guo, and Xiangqun Chen. 2017. Mining user reviews for mobile app comparisons. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3).

Nianzu Ma, Sahisnu Mazumder, Hao Wang, and Bing Liu. 2020. Entity-aware dependency-based deep graph attention network for comparative preference classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5782–5788, Online. Association for Computational Linguistics.

Walid Maalej and Hadeer Nabil. 2015. Bug report, feature request, or simply praise? On automatically classifying app reviews. In *Proceedings of the 23rd IEEE International Requirements Engineering Conference (RE)*, pages 116–125, Ottawa, ON, Canada. IEEE Press.

Fabio Palomba, Mario Linares-Vásquez, Gabriele Bavota, Rocco Oliveto, Massimiliano Di Penta, Denys Poshyvanyk, and Andrea De Lucia. 2015. User reviews matter! Tracking crowdsourced reviews to support evolution of successful apps. In *Proceedings of the 31st IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 291–300, Bremen, Germany. IEEE Press.

Alexander Panchenko, Alexander Bondarenko, Mirco Franzek, Matthias Hagen, and Chris Biemann. 2019. Categorizing comparative sentences. In *Proceedings* *of the 6th Workshop on Argument Mining*, pages 136–145, Florence, Italy. Association for Computational Linguistics.

Manny Rayner and Amelie Banks. 1988. Parsing and interpreting comparatives. In *26th Annual Meeting of the Association for Computational Linguistics*, pages 49–60, Buffalo, New York, USA. Association for Computational Linguistics.

Manny Rayner and Amelie Banks. 1990. An implementable semantics for comparative constructions. *Computational Linguistics*, 16(2):86–112.

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

Steffen Staab and Udo Hahn. 1997. Comparatives in context. In *Proceedings of the 14th National Conference on Artificial Intelligence*, AAAI/IAAI, page 616–621, Providence, Rhode Island. AAAI Press.

Arnim Von Stechow. 1984. Comparing semantic theories of comparison. *Journal of Semantics*, 3(1-2):1–77.

Robert van Rooij. 2011. Implicit versus explicit comparatives. In *Vagueness and Language Use*, pages 51–72. Palgrave Macmillan UK, London.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R. Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32, Vancouver. Curran Associates, Inc.

Counterfactual Explanations for Natural Language Interfaces

George Tolkachev
University of Pennsylvania
georgeto@seas.upenn.edu

Stephen Mell
University of Pennsylvania
sm1@seas.upenn.edu

Steve Zdancewic
University of Pennsylvania
stevez@seas.upenn.edu

Osbert Bastani
University of Pennsylvania
obastani@seas.upenn.edu

Abstract

A key challenge facing natural language interfaces is enabling users to understand the capabilities of the underlying system. We propose a novel approach for generating explanations of a natural language interface based on semantic parsing. We focus on counterfactual explanations, which are post-hoc explanations that describe to the user how they could have minimally modified their utterance to achieve their desired goal. In particular, the user provides an utterance along with a demonstration of their desired goal; then, our algorithm synthesizes a paraphrase of their utterance that is guaranteed to achieve their goal. In two user studies, we demonstrate that our approach substantially improves user performance, and that it generates explanations that more closely match the user's intent compared to two ablations.[1]

1 Introduction

Semantic parsing is a promising technique for enabling natural language user interfaces (Ge and Mooney, 2005; Artzi and Zettlemoyer, 2013; Berant et al., 2013; Wang et al., 2015). However, a key challenge facing semantic parsing is the richness of human language, which can often encode concepts (e.g., "circle") that do not exist in the underlying system or are encoded using different language (e.g., "ball"). Thus, human users can have trouble providing complex compositional commands in the form of natural language to such systems.

One approach to addressing this issue is to develop increasingly powerful models for understanding natural language (Gardner et al., 2018; Yin and Neubig, 2018). While there has been enormous progress in this direction, there remains a wide gap between what these models are capable of compared to human understanding (Lake and Baroni, 2018), manifesting in the fact that these models can fail in unexpected ways (Ribeiro et al., 2016). This gap can be particularly problematic for end users who do not understand the limitations of machine learning models, since it encourages the human user to provide complex commands, but then performs unreliably on such commands.

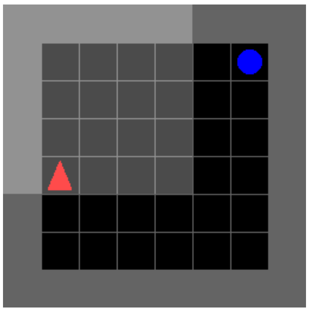
Thus, an important problem is to devise techniques for explaining these models. Generally speaking, a range of techniques have recently been developed for explaining machine learning models. The first technique is to use models that are intrinsically explainable, such as linear regression or decision trees. However, in the case of semantic parsing, such models may achieve suboptimal performance, and furthermore it is not clear that the structure of these models would be useful to end users. A second technique is to train a blackbox model, and then approximate it using an interpretable model. Then, the interpretable model can be shown to the human user to explain the high-level decision-making process underlying the blackbox model. However, this approach also suffers from the fact that showing a decision tree or regression model is likely not useful to an end user.

Instead, we consider an alternative form of explanation called a *counterfactual explanation* (Wachter et al., 2017). These explanations are designed to describe alternative outcomes to the user. In particular, given a prediction for a specific input, they tell the user how they could have minimally modified that input to achieve a different outcome. As an example, suppose a bank is using a machine learning model to help decide whether to provide a loan to an individual; if that individual is denied the loan, then the bank can provide them with a counterfactual explanation describing how they could change their covariates (e.g., increase their income) to qualify for a loan.

We propose a novel algorithm for computing counterfactual explanations for semantic parsers. In particular, suppose that a user provides a com-

[1] Code available at: https://github.com/georgeto20/counterfactual_explanations.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 113 - 118
May 22-27, 2022 ©2022 Association for Computational Linguistics



User command 1: “Go to the blue circle”
User command 2: “Go to the top right”
Our explanation: “Go to the blue ball”

Figure 1: Example BabyAI task (from Chevalier-Boisvert et al. (2018)), utterances, and our explanation.

mand in the form of a natural language utterance. If the natural language interface fails to provide the desired result, then our goal is to explain how the user could have modified their utterance to achieve the desired result. To this end, we have the human additionally provide the desired result. Then, we compute an alternative utterance that the semantic parser correctly processes while being as similar as possible to the original utterance. Intuitively, this explanation enables the user to modify their language to reliably achieve their goals in future interactions with the system.

We evaluate our approach on the BabyAI environment (Chevalier-Boisvert et al., 2018), where the human can provide a virtual agent with commands to achieve complex tasks such as “pick up the green ball and place it next to the blue box”. We perform two user studies, which demonstrate that our approach both produces correct explanations (i.e., match the user’s desired intent), and that it substantially improves the user’s ability to provide valid commands.

Example. In Figure 1, we show an example of a BabyAI task along with a user-provided utterance commanding the agent to go to the blue ball. The first command corresponds to a valid program, but cannot be understood by the semantic parser due to the use of the terminology “circle” instead of “ball”. The second command uses the construct “top right” that does not exist in the language. In both cases, the user provides a demonstration where the agent navigates next to the blue ball, upon which our approach generates the explanation shown.

Related work. There has been a great deal of recent interest in providing explanations of black-box machine learning models, focusing on explaining why the model makes an individual prediction (Ribeiro et al., 2016; Lei et al., 2016; Ribeiro et al., 2018; Alvarez-Melis and Jaakkola, 2018; Liu et al., 2018), or achieving better understanding of the limitations of models (Wallace et al., 2019; Ribeiro et al., 2020). In contrast, our goal is to explain how the input can be changed to achieve a desired outcome, which is called a counterfactual explanation (Wachter et al., 2017; Ustun et al., 2019). There has been interest in improving the performance of semantic parsers through interaction (Wang et al., 2016, 2017); our approach is complementary to this line of work, since it aims to make the system more transparent to the user. There has also been work on leveraging natural language descriptions to help generate counterfactual explanations for image classifiers (Hendricks et al., 2018), but not tailored at counterfactual predictions for natural language tasks; specifically, while their approach produces counterfactual explanations in natural language, they are for image predictions rather than text predictions.

For natural language processing tasks, a key challenge is that the input space is discrete (e.g., a natural language utterance); for such settings, there has been work on algorithms for searching over combinatorial spaces of counterfactual explanations (Ross et al., 2021b; Wu et al., 2021; Ross et al., 2021a). However, even for these approaches, the output space is typically small (e.g., a binary sentiment label). In contrast, semantic parsing has highly structured outputs (i.e., programs), requiring significantly different search procedures to find an explanation that produces the correct output. To address this challenge, we define a search space over counterfactual explanations for semantic parsing such that search is tractable.

2 Algorithm

Problem formulation. We consider the problem of computing counterfactual explanations for a semantic parsing model $f_\theta : \Sigma^* \to \Pi$. In particular, we assume the user provides a command in the form of an utterance $s \in \Sigma^*$, with the goal of obtaining some denotation $y \in \mathcal{Y}$. To achieve the user’s goal, the semantic parsing model produces a program $\pi = f_\theta(s) \in \Pi$, and then executes the program to obtain denotation $y = [\![\pi]\!] \in \mathcal{Y}$, where $[\![\cdot]\!] : \Pi \to \mathcal{Y}$ (called the *semantics* of Π) maps programs to outputs.

In this context, our goal is to provide explanations to the user to help them understand what utterances can be correctly understood and executed by the underlying system. In particular, we assume the user has provided an utterance s_0, but the output $[\![f_\theta(s_0)]\!]$ is not the one that they desired. Then, we ask the user to provide their desired output, after which we provide them with an alternative utterance s^* that is semantically similar to s_0 but successfully achieves y_0. Formally:

Definition 2.1. Given an utterance $s_0 \in \Sigma^*$ and a desired output $y_0 \in \mathcal{Y}$, the *counterfactual explanation* for s_0 and y_0 is the sentence

$$s^* = \arg\min_{s \in L} d(s, s_0) \text{ subj. to } [\![f_\theta(s)]\!] = y_0,$$

where d is a semantic similarity metric and $L \subseteq \Sigma^*$ is the search space of possible explanations.

The goal is that examining s^* should help the user provide utterances that are more likely to be correctly processed in future interactions.

Search space of explanations. A key challenge in generating natural language expressions is how to generate expressions that appear natural to the human user. To ensure that our explanations are natural, we restrict to sentences generated by a context-free grammar (CFG) C. In particular, we consider explanations in the form of sentences $s \in L(C) \subseteq \Sigma^*$ (where Σ is the vocabulary and $L(C)$ is the language generated by C). We restrict to sentences with parse trees of bounded depth d in C; we denote this subset by $L_d(C)$. In addition, we assume sentences $s \in L_d(C)$ are included in the dataset used to train the semantic parser f_θ to ensure it correctly parses these sentences.

Semantic similarity. Our goal is to compute a sentence $s \in L_d(C)$ that is semantically similar to the user-provided utterance s_0. To capture this notion of semantic similarity, we use a pretrained language model $x = g_\theta(s)$ that maps a given sentence s to a vector embedding $x \in \mathbb{R}^k$. Then, we use cosine similarity in this embedding space to measure semantic similarity. In particular, we use the distance $d(s, s_0) = 1 - \text{sim}(g_\theta(s), g_\theta(s_0))$, where $\text{sim}(x, x')$ is the cosine similarity.

Goal constraint. Finally, we want to ensure that the provided explanation successfully evaluates to the user's desired denotation y_0. For a given utterance s, we can check this constraint simply by evaluating $y = [\![f_\theta(s)]\!]$ and checking if $y = y_0$.

Algorithm 1 Our algorithm for computing counterfactual explanations for a semantic parser f_θ.

```
procedure EXPLAIN($s_0, y_0$)
    $(s^*, c^*) \leftarrow (\varnothing, -\infty)$
    for $s \in L_d(C)$ do
        if $[\![f_\theta(s)]\!] = y_0$ then
            $c \leftarrow \text{sim}(g_\theta(s), g_\theta(s_0))$
            if $c > c^*$ then $s^*, c^* \leftarrow s, c$ end if
        end if
    end for
    return $s^*$
end procedure
```

Overall algorithm. Given user-provided utterance s_0 and desired denotation y_0, the counterfactual explanation problem is equivalent to:

$$s^* = \arg\max_{s \in L_d(C)} \text{sim}(g_\theta(s), g_\theta(s_0)) \quad \text{subj. to } [\![f_\theta(s)]\!] = y_0.$$

Assuming $L_d(C)$ is sufficiently small, we can solve this problem by enumerating through the possible choices $s \in L_d(C)$ and choosing the highest scoring one that satisfies the constraint. In practice, we may be able to exploit the structure of the constraint to prune the search space. Our approach is summarized in Algorithm 1.

3 Experiments

We perform two user studies to demonstrate (i) correctness: our explanations preserve the user's original intent, and (ii) usefulness: our explanations improve user performance.

3.1 BabyAI Task

We evaluate our approach on BabyAI (Chevalier-Boisvert et al., 2018) adapted to our setting. In this task, the human can provide commands to an agent navigating a maze of rooms containing keys, boxes, and balls. The goal is defined by the combination of the agent position and the environment state (e.g., the agent may need to place a ball next to a box). Atomic commands (e.g., going to, picking up, or putting down an object) can then be composed in sequence to achieve complex goals. In our setup, s_0 is a natural language command, and y_0 is a demonstration in the form of a trajectory the agent could take to achieve the desired goal.

This task comes with a context-free grammar of natural language commands, which we use as the space of possible explanations. Next, we train a semantic parser to understand commands from this grammar. Since utterances in this grammar correspond one-to-one with programs, we can generate training data. We generate 1000 training examples (s, π) consisting of an utterance s along with a program π, and train TranX (Yin and Neubig, 2018) to predict $\pi = f_\theta(s)$. For semantic similarity, we use a pretrained DistilBERT model g_θ (Devlin et al., 2018; Sanh et al., 2019) to embed utterances s.

Handling the goal constraint is more challenging, since the denotation can be nondeterministic—in particular, multiple different trajectories can be used to achieve a single goal (e.g., there are multiple paths the agent can take to a given object). Thus, if we naïvely take the denotation of a program to be a single trajectory that achieves the goal, then this trajectory may be different than the given demonstration even if the demonstration also achieves the goal. To address this issue, we instead enumerate the set Π_y of all possible programs that are consistent with the given demonstration y, up to a bounded depth (selected so that Π_y is large enough while ensuring that the experiments still run quickly). Then, we replace the constraint $[\![f_\theta(s)]\!] = y_0$ with a constraint saying that $f_\theta(s)$ is in this set—i.e., $f_\theta(s) \in \Pi_{y_0}$.

3.2 Correctness of Explanations

We evaluate whether our explanations are valid paraphrases of the user's original command.

Baselines. We compare to two ablations of our algorithm. The first one omits the goal constraint $f_\theta(s) \in \Pi_y$; thus, it simply returns the explanation that is most semantically similar to the user-provided utterance s_0. Intuitively, this ablation evaluates the usefulness of the goal constraint.

The second ablation ignores s_0, and returns an explanation s such that $f_\theta(s) \in \Pi_{y_0}$; we choose s to minimize perplexity according to GPT-2. Intuitively, this ablation measures the usefulness of specializing the explanation to the user's utterance.

Setup. We selected 17 BabyAI tasks by randomly sampling BabyAI levels until we obtain a set of tasks of varying difficulty. For example, Task 1 has the simple goal "go to the green ball", while Task 10 has the more complex goal "pick up a green key, then put the yellow box next to the grey ball".

Then, our experiment proceeds in two phases. In the first phase, we use Amazon Mechanical Turk (AMT) to collect natural language commands for the agent. For each of our 17 tasks, we show the user a video of the BabyAI agent achieving the task, and then ask them to provide a single command that encodes the goal. In total, we obtained 127 commands (one per user) for each task. Next, for each user instruction, we find the counterfactual explanation according to our algorithm and the two ablations described above.

Approach	Correctness	Usefulness
Ours	41.4 ± 1.48%	50.8 ± 2.22%
No demo	34.0 ± 1.42%	49.2 ± 2.01%
GPT-2	24.6 ± 1.29%	46.2 ± 1.96%
No training	–	10.2 ± 0.74%

Table 1: Correctness: The frequency at which users chose the explanation generated using the corresponding approach as the best match. Usefulness: The percentage of user utterances correctly parsed (averaged across the last 10 tasks), where users are given explanations generated by the corresponding approach.

In the second phase, we conduct a second AMT study to evaluate the correctness of these explanations. In particular, for each of our 17 tasks, we show each participant a single command for that task (chosen randomly from the 127 commands in the first phase), along with the three generated explanations and the video of the agent achieving that task. Then, we ask the user to choose the explanation that is closest in meaning to the original command. We obtained 50 responses.

Results. In Table 1, we show the fraction of times users in the second phase selected each explanation, averaged across both users and tasks. Our approach significantly outperforms GPT-2, which is unsurprising since this ablation makes no effort to preserve the user's intent. Our approach also outperforms the ablation without the goal constraint, demonstrating the usefulness of this constraint.

3.3 Usefulness of Explanations

Next, we evaluate whether providing explanations can make it easier for users to provide commands that can be understood by our semantic parser.

Baselines. In addition to the two ablations in Section 3.2, we also compare to a baseline where the user is not provided with any explanation.

Setup. We run an AMT study similar to the first phase of our study in Section 3.2, except immediately after providing a command for a task, each user is shown an explanation for their command and that task. We collected 50 user responses.

Results. For each user command s_0, we run our semantic parser to obtain the corresponding program and check whether it is in the set of programs valid for that task—i.e., whether $f_\theta(s_0) \in \Pi_{y_0}$. Table 1 shows the success rate across all users and the last 10 tasks; we restrict to the last 10 to give the user time to learn to improve their performance. Users not provided any explanations performed very poorly overall. The remaining approaches performed similarly; our explanations led to the best performance, followed closely by the ablation without the demonstration, with a wider gap to the ablation that ignores the user utterance. Thus, personalizing the explanation to the user based on their utterance helps improve performance.

4 Conclusion

We have proposed a technique for explaining how users can adapt their utterances to interact with a natural language interface. Our experiments demonstrate how our explanations can be used to significantly improve the usability of semantic parsers when they are limited in terms of their semantic understanding. While any explanations are already very useful, we show that personalizing explanations can further improve performance.

A key design choice in our approach is to construct a synthetic grammar from which counterfactual explanations are generated. In a realistic application, the semantic parsing model can be trained on a combination of synthetic data and real-world data, enabling our approach to be used in conjunction with the synthetic grammar. A key direction for future work is extending our approach to settings where such a grammar is not available. In our experience, a key challenge in this setting is that the generated text can be unnatural, possibly due to the constraints imposed on the search space.

Acknowledgments

We gratefully acknowledge support from NSF CCF-1917852 and CCF-1910769. The views expressed are those of the authors and do not reflect the official policy or position of the U.S. Government.

References

David Alvarez-Melis and Tommi S Jaakkola. 2018. Towards robust interpretability with self-explaining neural networks. *arXiv preprint arXiv:1806.07538*.

Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1:49–62.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. 2018. Babyai: A platform to study the sample efficiency of grounded language learning. *arXiv preprint arXiv:1810.08272*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Matt Gardner, Pradeep Dasigi, Srinivasan Iyer, Alane Suhr, and Luke Zettlemoyer. 2018. Neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 17–18.

Ruifang Ge and Raymond Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 9–16.

Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. 2018. Generating counterfactual explanations with natural language. *arXiv preprint arXiv:1806.09809*.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pages 2873–2882. PMLR.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

Hui Liu, Qingyu Yin, and William Yang Wang. 2018. Towards explainable nlp: A generative explanation framework for text classification. *arXiv preprint arXiv:1811.00196*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of nlp models with checklist. *arXiv preprint arXiv:2005.04118*.

Alexis Ross, Himabindu Lakkaraju, and Osbert Bastani. 2021a. Learning models for actionable recourse. *Advances in Neural Information Processing Systems*, 34.

Alexis Ross, Ana Marasović, and Matthew E Peters. 2021b. Explaining nlp models via minimal contrastive editing (mice). In *Findings of the ACL*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Berk Ustun, Alexander Spangher, and Yang Liu. 2019. Actionable recourse in linear classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 10–19.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. *arXiv preprint arXiv:1908.07125*.

Sida I Wang, Samuel Ginn, Percy Liang, and Christoper D Manning. 2017. Naturalizing a programming language via interactive learning. *arXiv preprint arXiv:1704.06956*.

Sida I Wang, Percy Liang, and Christopher D Manning. 2016. Learning language games through interaction. *arXiv preprint arXiv:1606.02447*.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1332–1342.

Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S Weld. 2021. Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models. In *ACL*.

Pengcheng Yin and Graham Neubig. 2018. Tranx: A transition-based neural abstract syntax parser for semantic parsing and code generation. *arXiv preprint arXiv:1810.02720*.

Predicting Difficulty and Discrimination of Natural Language Questions

Matthew A. Byrd   **Shashank Srivastava**
University of North Carolina at Chapel Hill
matthew_a_byrd@outlook.com, ssrivastava@cs.unc.edu

Abstract

Item Response Theory (IRT) has been extensively used to numerically characterize question difficulty and discrimination for human subjects in domains including cognitive psychology and education (Primi et al., 2014; Downing, 2003). More recently, IRT has been used to similarly characterize item difficulty and discrimination for natural language models across various datasets (Lalor et al., 2019; Vania et al., 2021; Rodriguez et al., 2021). In this work, we explore predictive models for directly estimating and explaining these traits for natural language questions in a question-answering context. We use HotpotQA for illustration. Our experiments show that it is possible to predict both difficulty and discrimination parameters for new questions, and these traits are correlated with features of questions, answers, and associated contexts. Our findings can have significant implications for the creation of new datasets and tests on the one hand and strategies such as active learning and curriculum learning on the other.

1 Introduction

The use of question answering for testing learning often relies on characterizing questions on aspects such as *difficulty* and *discrimination*[1]. For example, ordering questions by difficulty can enable curriculum learning (Bengio et al., 2009). Similarly, discrimination is used in standardized exams such as the SAT to ensure that questions are varied enough to discriminate between high-ability and low-ability respondents. Item Response Theory (IRT) (Wright and Stone, 1979; Lord, 1980) has been a widely applied framework to jointly estimate such parameters for questions (or *items*) and the abilities of *respondents*. While IRT has its inception in psychometrics and has traditionally been used with human respondents, recently, it has been explored for analyzing predictions from an 'artificial crowd' of ML models (Prudêncio et al., 2015; Plumed et al., 2016; Martínez-Plumed et al., 2019; Lalor et al., 2019; Vania et al., 2021; Rodriguez et al., 2021).

While it can be helpful to know which questions are difficult/discriminatory, it can be equally important to be able to determine a question's difficulty/discrimination parameters without having to use it in a testing environment (as is required to estimate IRT parameters). Some recent work, such as Ha et al. (2019), has explored using features derived from the text of a question to predict the difficulty in the context of multiple-choice medical exams. While others (Benedetto et al., 2020) have used tf-idf features to predict the difficulty of questions as measured by IRT. We differ from these works in two ways: Firstly, while Ha et al. (2019); Benedetto et al. (2020) both predict the difficulty of items for humans, we are interested in predicting the difficulty (and discrimination) of items for QA models. Secondly, we choose a question-answering dataset, HotpotQA (Yang et al., 2018), as our testbed. We utilize this dataset to generate a rich and varied feature set across each item's question, answer, and associated contexts. We can then employ these features to analyze our difficulty and discrimination predictions, giving us insights into both our underlying QA model and factors that can increase the difficulty/discrimination of a question.

Our analysis shows significant variations among questions and reveals some surprising patterns. We show that it is possible to predict both difficulty and discrimination of natural language questions, which can have multiple applications in education and pedagogy. Additionally, we see that different surface-level features are associated with high discrimination and high difficulty, which can inform

[1] By difficulty, we refer to how likely a respondent is to answer a question correctly, whereas by discrimination we refer to the value of a question in identifying a given level of ability in respondents. A question like '$2 + 2 = ?$' has low difficulty but potentially high discrimination, since a respondent who answers incorrectly is likely to have no arithmetic ability.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 119 - 130
May 22-27, 2022 ©2022 Association for Computational Linguistics

new evaluation methods and the creation of new datasets. Further, we identify attributes for predicting difficulty and discrimination that are general enough to be adapted to various QA datasets.[2]

2 IRT Analysis of HotpotQA

IRT background: We begin by summarizing the 1PL and 2PL models from IRT, which form the basis of our later analysis. The 1PL (1 Parameter Logistic) model describes the probability of respondent i correctly answering the j'th item (question) in terms of scalar-valued parameters for question difficulty (d_j) and respondent ability (θ_i). These parameters are estimated from data $y_{ij} \in \{0, 1\}$ for a set of i, j pairs. Here, $y_{ij} = 1$ indicates a correct answer. The 1PL model is described by:

$$p(y_{ij} = 1|\theta_i, d_j) = \frac{1}{1 + e^{-(\theta_i - d_j)}}$$

The 2PL model extends the 1PL by adding a scalar-valued parameter α_j, which represents the discrimination of the j'th item. Intuitively, this parameter denotes how sharply the probability of answering a question correctly changes as the ability of the respondent increases. The 2PL model is described by:

$$p(y_{ij} = 1|\theta_i, d_j, \alpha_j) = \frac{1}{1 + e^{-\alpha_j(\theta_i - d_j)}}$$

Dataset description: We chose HotpotQA for our analysis since it is significantly more complex than other datasets such as SQuAD (Rajpurkar et al., 2016) due to the questions requiring multi-hop reasoning and having more complex language. In HotpotQA, each question is paired with two paragraphs considered 'gold' contexts and several other paragraphs considered 'distractor' contexts. The answer to each question is a span in one of the gold contexts, but correctly answering the question requires combining information from both 'gold' contexts.

2.1 Estimating IRT Parameters

We estimate the IRT parameters for the questions in HotpotQA's dev set (7,405 questions). However, collecting human responses for each question, which is necessary to estimate IRT parameters, is infeasible. Motivated by Lalor et al. (2019), we create an artificial crowd of QA models in place of a crowd of human respondents. For this, we train 148 instances of DFGN (Qiu et al., 2019) models on HotpotQA's train set.[3] To ensure diversity, we uniformly sample the number of training epochs from 1 to 15 and sample the fraction of the training data used for model training from $\mathcal{U}(0, 1)$. Otherwise, each model was trained with the hyperparameters described in Qiu et al. (2019). Next, we generate an *item-response matrix* indicating which questions from the HotpotQA dev set each model answered correctly (i.e., the model's answer exactly matched the correct answer). We remove any questions that received no correct answers or no incorrect answers. This is done as during the estimation process, these questions tend towards (+/-) infinity in their difficulty parameters, as well, their discrimination parameter estimate tends towards zero (unable to distinguish between high and low performing models). Our final dataset is a subset of 4,000 questions (2,000 train, 1,000 dev, and 1,000 test). Finally, we fit the 1PL and 2PL models on the foresaid item-response matrix using the variational IRT training procedure from Natesan et al. (2016).

2.2 Analysis of Estimated Parameters

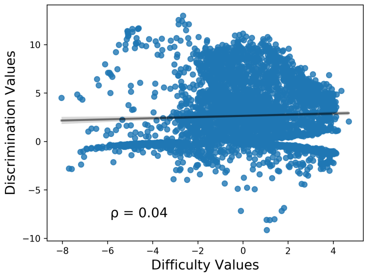


Figure 1: 2PL discrimination vs 1PL difficulty for questions.

Figure 1 shows a scatter-plot of estimated difficulty and discrimination values for individual questions. We note that some discrimination values asymptotically approach 0. This occurs when some questions receive very few or many correct answers; these questions cannot discriminate high-performing from low-performing models. We also note that some questions have negative discrimination, i.e., as a model's ability increases, its probability of answering the question correctly decreases. This is primarily a result of some of the highest per-

[2] Code, models, and data for all experiments are available at https://github.com/ByrdOfAFeather/pred_irt

[3] We choose DFGN due to its competitive performance on the HotpotQA leaderboard, the number of models we train is primarily driven by computational limits.

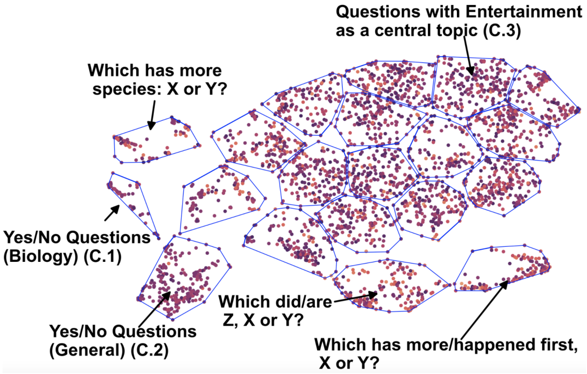


Figure 2: All 3000 questions from our train/dev set as UMAP-reduced BERT embeddings, color-coded by difficulty (darker is more difficult). We find that clusters produced by KMeans ($K = 20$) naturally cluster together questions that are similar in how they are asked or topics that are asked about. We label some clusters according to these types. We specially mark C.1, C.2, and C.3. C.1 and C.2 have uniformity in the type of question being asked, as well as lower variance than other clusters. C.3 is uniform in topic but can vary in the type of question.

forming models giving an answer which is either a subspan of or contains the ground-truth answer of questions that were otherwise answered correctly by lower-performing models. Overall, there is a weak positive correlation between discrimination and difficulty ($\rho = 0.04$).

To visualize any correlation between the semantic and syntactic information of questions and their respective difficulty levels, we clustered questions based on their BERT embeddings using KMeans ($K = 20$) clustering (2D UMAP reduction shown in Figure 2). Through manually examining and labeling the clusters, we found that many clusters could be described with a specific style (e.g., yes/no questions) or general topic. Some clusters, such as C.3, have a large variety in the phrasing of questions being asked and the potential answers in both syntactic and semantic features. For example, both *Q: Khushi Ek Roag is broadcast by a company based out of where? A: Dubai* and *Q: To Catch a Predator was devoted to impersonating people below the age of consent for which in North America varies by what? A: jurisdiction* are in C.3.

Other clusters, such as C.1 and C.2, (yes/no clusters), only vary in topic rather than the type of question. In particular, for these clusters, the estimated difficulty has significantly lower variance than the other clusters ($\rho = 0.02$, $\rho = 0.04$ respectively), indicating that these yes/no questions tend to be consistent in their difficulty. The standard deviation values for C.1 and C.2 are 1.08 and 1.19 respectively, the average standard deviation value is 2.27. We further explore how these factors affect predicting the difficulty values in section 4.

3 Predicting IRT Parameters

We next discuss predictive models for discrimination and difficulty using features from the question, answer, and associated context. First, we describe our feature set, then provide an ablation study, a feature importance study, and finally qualitatively analyze the predictions of our best model.

3.1 Feature Design

We experiment with two categories of features: human-centric and machine-centric features. For human-centric features, we considered (1) counting-based **Lexical & Syntactic features** extracted for both questions and answers like ContentWords, Type-token ratio, Avg. Word Length, Complex Words ($>$ 3 syllables); (2) **Semantic-Ambiguity features** measuring a question's or answer's ambiguity (Ha et al., 2019); and (3) **Readability features** based on measures like Fleisch Kincaid index. More feature details can be found in Appendix C. For machine-centric features, we considered (1) **Contextual Embeddings** for questions and answers from BERT (Devlin et al., 2019); (2) n-gram **Overlap Counts** between the question and answer, and between question/answer and the gold/distractor paragraphs; and (3) **POS Counts** from the Stanford Tagset (Toutanova et al., 2003) for the question and answer.

3.2 Quantitative Analysis and Ablation

Table 1 and Table 2 show the regression performance of our models for predicting the IRT difficulty/discrimination parameters of the questions in our dev/test sets using the feature sets described before. The reported results are averaged over a 10-fold cross-validation. We note that the best models for both difficulty and discrimination show significant ($\rho < 0.10$) predictive performance (R^2 of 0.17 and 0.13) against our baseline (Mean).

The best performance is achieved in both tasks by considering all features. In both cases, there is a significant difference ($\rho < 0.1$) in performance between using any single set and using all features, except the best-performing BERT feature set. We also note that features derived from the answer are typically better at capturing difficulty, while features derived from the question better predict the discrimination parameters. However, the per-

Features	Dev MSE	Dev R^2	Test MSE	Test R^2
All	**5.14**	**0.11**	**4.72**	**0.17**
All (Q)	5.43	0.07	5.10	0.10
All (A)	5.41	0.08	5.05	0.11
BERT (Q)	5.41	0.07	4.99	0.12
BERT (A)	5.25	0.10	5.05	0.11
H.C. (Q)	5.62	0.01	5.38	0.05
H.C. (A)	5.45	0.06	5.20	0.08
Lex. & Syn. (Q)	5.62	0.01	5.37	0.05
Lex. & Syn. (A)	5.47	0.03	5.36	0.06
Read. (Q)	5.80	0.00	5.71	0.00
Read. (A)	5.63	0.02	5.48	0.03
Sem. Ambiguity (Q)	5.76	0.01	5.55	0.02
Sem. Ambiguity (A)	5.81	0.01	5.68	0.00
P.O.S. (Q)	5.37	0.05	5.23	0.08
P.O.S. (A)	5.60	0.01	5.28	0.07
A/Q/C Overlap	5.39	0.05	4.92	0.13
Mean	5.82	0.00	5.69	0.00

Table 1: Results for predicting the 1PL difficulty parameters. BERT (Q) and BERT (A) use the BERT embeddings for the question/answer respectively. H.C. (Q)/(A) are the human-centric features for the question/answer respectively. A/Q/C Overlap is using only the overlap counts between question, answer, and contexts.

formance of All (Q) and All (A) for both the discrimination and difficulty is weaker than using all features. Since the difference is not statistically significant, it is unclear how much predictive power is added when considering both answer and question features in these predictions.

The features that focus on human difficulty are among the less effective feature sets, indicating that the human difficulty features of a question do not fully capture difficulty for QA models. We provide details of models and their training and the experiment setup in Appendix A; as well, significance tests can be found in Appendix D.

3.3 Feature Importance Study

We estimated feature importance by permuting each feature individually and measuring the change in MSE on the dev set. We list features that caused a change in MSE of at least .01 in tables 3 and 4.

We point out that for predicting the discrimination, the number of cardinal digits in the answer was the most important indicator of high discrimination. The positive correlation between the number of digits in the answer and the discrimination of a question is expected. Qiu et al. (2019) showed that the DFGN model has a significant weakness in numeric operations. This gives questions with numeric answers a high discrimination value as DFGN models are naturally inhibited in this regard, and thus only a few models with the most training data will be capable of answering these questions. We find a similar positive Pearson score ($\rho = 0.14$) between the difficulty and the number of cardinal digits in the answer. While this weakness of the DFGN model cannot be applied to an arbitrary QA model, the methodology used to determine this weakness can be applied arbitrarily, which can give solid grounding to claims about model weaknesses.

Features	Dev MSE	Dev R^2	Test MSE	Test R^2
All	9.08	**0.13**	**9.14**	**0.13**
All (Q)	9.32	0.10	9.50	0.09
All (A)	9.59	0.08	9.98	0.04
BERT (Q)	**9.02**	0.11	9.27	0.11
BERT (A)	9.52	0.08	9.64	0.08
H.C (Q)	9.76	0.04	9.86	0.06
H.C (A)	10.09	0.03	10.31	0.02
Lex. & Syn. (Q)	9.75	0.04	9.86	0.06
Lex. & Syn. (A)	10.13	0.01	10.21	0.03
Read. (Q)	10.08	0.01	10.17	0.03
Read. (A)	10.13	0.02	10.31	0.01
Sem. Ambiguity (Q)	10.05	0.02	10.16	0.03
Sem. Ambiguity (A)	10.21	0.00	10.47	0.00
P.O.S. (Q)	9.96	0.04	10.10	0.03
P.O.S. (A)	9.78	0.03	9.82	0.06
A/Q/C Overlap	9.56	0.06	9.63	0.08
Mean	10.21	0.00	10.53	0.00

Table 2: Results for predicting the 2PL discrimination parameters. The setup is the same as in table 1. BERT (Q) has the highest performance. However, the difference in performance when using BERT (Q) compared to using All is not statistically significant. See Appendix D for significance tests.

Feature	Change in MSE	Interval	Corr.
# Commas A.	0.06	± 0.02	0.10
# Complex Words A.	0.05	± 0.01	-0.04
# NNP A.	0.05	± 0.02	-0.16
# SNP A/G.C.	0.02	± 0.01	0.04
# Commas Q.	0.01	± 0.01	-0.11

Table 3: Feature importances for difficulty parameters (all features considered). A. refers to a feature capturing information from the answer, Q. refers to a feature capturing information from the question. A/G.C. refers to a feature measuring overlap between the answer and gold contexts.

Feature	Change in MSE	Interval	Corr.
# CD A.	0.25	± 0.03	0.17
# Commas Q.	0.08	± 0.02	-0.11
Avg. Sense/Adverb A.	0.01	± 0.02	-0.03

Table 4: Feature importances for discrimination parameters (all features considered)

4 Qualitative Analysis

We qualitatively analyze the difficulty predictions to understand the predictions of our best-performing model. Similar to Figure 2, Figure 3 shows a UMAP scatterplot[4] for questions on our test split of the estimated IRT parameters. In this case, instead of color-coding by difficulty as in Figure 2, we instead color-code by the absolute error between our predictions and the measured difficulty of each question. We again apply KMeans ($k = 10$) to our data with a smaller number of clusters due to the smaller size of the test set. We highlight CT.1, like C.1 and C.2 of Figure 2, this cluster consists primarily of yes/no questions. The difficulty in CT.1 has significantly smaller variance in the estimated difficulties than the rest of the clusters ($\rho = 0.02$). As well, the prediction error for CT.1 has significantly smaller variance ($\rho = 0.04$) and had the smallest average prediction error compared to the other clusters (0.68). This indicates that the model is able to recognize when question groupings, such as yes/no questions, have consistent difficulties (as discussed in 2.1) and has consistently lower error when predicting difficulty for these questions. However, the prediction error tends to vary more when the surface-level question types are not sufficient to characterize their difficulty.

We explore this further through a small counterfactual experiment. We are interested in taking an item with high prediction error and slightly tweaking it to understand how the model's predictions can change with changes in the question and answer. We selected an item with >2 absolute error to perform this experiment. The question we use in this study is: *Which university is this American philosopher, theologian, and Christian apologist who supports theistic science, professor at?* with an answer of *Biola University*. The predicted difficulty was -0.51. We found that simple changes to the question, such as using synonyms and removing unnecessary information, can increase the predicted difficulty up to -0.21. However, by modifying the answer (and by necessity the question) to be either a date or yes, we achieve a higher difficulty prediction (0.53 and 1.02, respectively). This further indicates the model's bias towards yes/no questions being of a higher difficulty regardless of the style or topic of question being asked. Some of our changes and their corresponding predictions are listed in Appendix E.

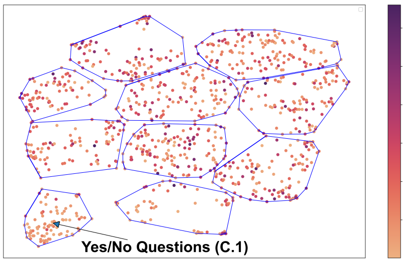


Figure 3: UMAP scatterplot of questions color coded by prediction error for difficulty. (Test set)

5 Conclusion

In this paper, we explored QA datasets through the lens of Item Response Theory. We have demonstrated a way to build regression models that can describe the difficulty and discrimination of a question. We note that our work is limited in two important ways: firstly, we only use the DFGN model in our artificial crowd, which may have introduced a bias in which some factors that make questions difficult/discriminatory are only applicable to this model. Secondly, we only explore the HotPotQA dataset, which may further limit our analysis to only be applicable to HotPotQA or similar datasets. Future work could incorporate multiple models and datasets to explore a more easily generalizable difficulty/discrimination prediction pipeline. We also note that our analysis here focused on QA. However, there are many NLP tasks in which the difficulty or discrimination of an item may be important. Our work here could naturally extend to these domains. Finally, automatically predicting these traits without relying on user responses can engender a host of creative educational applications. Future work can also leverage such predictive models to explore more efficient strategies for learning and evaluation.

References

Moez Ali. 2020. PyCaret: An open source, low-code machine learning library in Python. PyCaret version 2.2.

Luca Benedetto, Andrea Cappelli, Roberto Turrin, and Paolo Cremonesi. 2020. R2DE: a NLP approach to estimating IRT parameters of newly generated questions. In Proceedings of the Tenth International Conference on Learning Analytics & Knowledge, pages 412–421.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert,

[4] Similar plots for the discrimination parameters are included in Appendix G

and Jason Weston. 2009. Curriculum learning. In Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, page 41–48, New York, NY, USA. Association for Computing Machinery.

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

Steven M Downing. 2003. Item response theory: applications of modern test theory in medical education. Medical Education, 37(8):739–745.

Eileen B. Entin and George R. Klare. 1978. Some inter-relationships of readability, cloze and multiple choice scores on a reading comprehension test. Journal of Reading Behavior, 10(4):417–436.

R. Flesch. A new readability yardstick. Journal of applied psychology, 32(3).

R. Gunning. 1952. The Technique of Clear Writing. McGraw-Hill, New York.

Le An Ha, Victoria Yaneva, Peter Baldwin, and Janet Mee. 2019. Predicting the difficulty of multiple choice questions in a high-stakes medical exam. In Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, pages 11–20, Florence, Italy. Association for Computational Linguistics.

J. Kincaid, R. P. Fishburne, R. L. Rogers, and B. S. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel.

John P Lalor, Hao Wu, and Hong Yu. 2019. Learning latent parameters without human response patterns: Item response theory with artificial crowds. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing.

G. Harry Mc Laughlin. 1969. SMOG grading-a new readability formula. Journal of Reading, 12(8):639–646.

Frederic M. Lord. 1980. Applications of Item Response Theory to Practical Testing Problems. Routledge.

Fernando Martínez-Plumed, Ricardo B.C. Prudêncio, Adolfo Martínez-Usó, and José Hernández-Orallo. 2019. Item response theory in ai: Analysing machine learning classifiers at the instance level. Artificial Intelligence, 271:18 – 42.

George A. Miller. 1995. WordNet: A lexical database for English. Commun. ACM, 38(11):39–41.

P Natesan, R Nandakumar, T Minka, and JD Rubright. 2016. Bayesian prior choice in irt estimation using mcmc and variational bayes. Front. Psychol. 7: 1422. doi: 10.3389/fpsyg.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830.

Fernando Plumed, Ricardo Prudêncio, Adolfo Martínez-Usó, and Jose Hernandez-Orallo. 2016. Making sense of Item Response Theory in machine learning.

Caterina Primi, Kinga Morsanyi, Maria Anna Donati, and Francesca Chiesi. 2014. Item Response Theory analysis of the Cognitive Reflection Test: Testing the psychometric properties of the original scale and a newly developed 8-item version, pages 2799–2804.

R. Prudêncio, J. Hernández-Orallo, and A. Martınez-Usó. 2015. Analysis of instance hardness in machine learning using item response theory.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 6140–6150, Florence, Italy. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text.

Pedro Rodriguez, Joe Barrow, Alexander Miserlis Hoyle, John P. Lalor, Robin Jia, and Jordan Boyd-Graber. 2021. Evaluation examples are not equally informative: How should that change NLP leaderboards? In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4486–4503, Online. Association for Computational Linguistics.

F. A. Smith and R.J. Senter. 1967. Automated readability index. Technical Report AMRL-TR-6620.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1, NAACL '03, page 173–180, USA. Association for Computational Linguistics.

Clara Vania, Phu Mon Htut, William Huang, Dhara Mungra, Richard Yuanzhe Pang, Jason Phang, Haokun Liu, Kyunghyun Cho, and Samuel R. Bowman. 2021. Comparing test sets with item response theory. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1141–1158, Online. Association for Computational Linguistics.

Benjamin D. Wright and Mark H. Stone. 1979. Best test design. Mesa Press.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Conference on Empirical Methods in Natural Language Processing (EMNLP).

A Models & Training

For the 1PL and 2PL prediction, we considered linear models with L1 & L2 regularization, random forests, gradient boosted regressors, and bayesian ridge models. All hyperparameters were kept constant as the default in the sklearn package (Pedregosa et al., 2011). We performed 10-fold cross-validation using PyCaret (Ali, 2020). All models were trained on a consumer grade processor.

B Feature Definitions

- **Human-Centric Features**
  - **Lexical & Syntactic features**: These consist primarily of counting features: ContentWords, Type-token ratio, Avg. Word Length, Complex Words (> 3 syllables). These are calculated for both the answer and question. A full list of these features can be found in Appendix F
  - **Semantic-Ambiguity features**: We use WordNet (Miller, 1995) to calculate the ambiguity of sentences, similar to Ha et al. (2019). These are calculated for both answer and question.
  - **Readability features**: We use previous work (Kincaid et al., 1975; Gunning, 1952; Laughlin, 1969) to model the readability of a question/answer (e.g. Fleisch Kincaid index). These are further expanded on in Appendix C.
- **Machine-Centric Features**
  - **Contextual Embeddings**: We use the BERT-base model (Devlin et al., 2019) to obtain sentence embeddings for questions and answers.
  - **Overlap Counts**: We count overlaps between the question and answer of n-grams up to $n = 3$. We also compute overlap counts between the question/answer and the gold and distractor paragraphs.
  - **Part of Speech Counts**: We count POS tags for tags from the Stanford NLP tagset (Toutanova et al., 2003) for both the question and answer.

C Reading Difficulty Features

We list the reading difficulty features we used in our experiments and an overview of their calculations. Each calculation has its own coefficients that can be found in their respective citations.

- Flesch Reading Ease - linear combination of words/sentence and syllables/word (Flesch)
- Flesch Kincaid Grade Level - linear combination of word/sentence and syllables/word (Kincaid et al., 1975)
- Automated Readability Index (ARI) - linear combination of characters/word and words/sentence (Smith and Senter, 1967)
- Gunning Fog index - linear combination of words/sentence and complex words/words. Complex words are words with 3 syllabus (Gunning, 1952)
- Coleman-Liau - linear combination of letters/100 words and sentences/100 words.(Entin and Klare, 1978)
- SMOG index - calculates the grade level by considering the number of complex words/sentence (Laughlin, 1969)

D Significance Tests

We provide significance tests for the difficulty and discrimination predictions in tables 5 and 6. We see that the BERT features and using all features are able to beat the baseline with statistical significance ($\rho \leq .1$). Note that we compare using MSE rather than R^2 as the baseline always has an R^2 score of 0. We also provide in table 7 the significance tests for using all features against BERT features. We find that the best performing BERT feature set does not have a statistically significant improvement in performance when compared to the all feature set. In this case, we use R^2 as the performance metric.

Features	**p**
All	**0.034**
BERT (Q)	0.211
BERT (A)	**0.078**
H.C. (Q)	0.551
H.C. (A)	0.261
A/Q Con.	0.674
P.O.S. (Q)	0.501
P.O.S. (A)	0.523

Table 5: 1PL difficulty predictions. P-values for feature set performance (MSE) tested against the baseline.

E Counterfactual Results

- - Question (original): Which university is this American philosopher, theologian, and Christian apologist, who supports theistic science, professor at?'

Features	p
All	**0.007**
BERT (Q)	**0.013**
BERT (A)	**0.098**
H.C. (Q)	0.165
H.C. (A)	0.726
A/Q Con.	0.831
P.O.S. (Q)	0.656
P.O.S. (A)	0.174

Table 6: 2PL discrimination predictions. P-values for feature set performance (MSE) tested against the baseline.

Features	p
BERT (Q) (Diff.)	**0.042**
BERT (Q) (Discrim.)	0.769
BERT (A) (Diff.)	0.278
BERT (A) (Discrim.)	**0.089**

Table 7: 1PL and 2PL Difficulty and Discrimination predictions. P-values for BERT performance (R^2) tested against all features performance.

  - Answer: "Biola University"
  - Pred. Diff: -0.51

- - Question : Which school is this philosopher and theologian who supports science, professor at?
  - Answer: "Biola University"
  - Pred. Diff: -0.21

- - Question : What was the birth date of a professor at Biola University who is an American philosopher, theologian, and Christian apologist, who supports theistic science?
  - Answer: March 9, 1948
  - Pred. Diff: 0.53

- - Question : Does Biola University have a professor who is an American philosopher, theologian, and Christian apologist, who supports theistic science?
  - Answer: yes
  - Pred. Diff: 1.02

F Lexical Features

We list our full list of lexical features, these features are a subset of the lexical features used in Ha et al. (2019).

- Word Count
- Content Word Count
- Content Word Incidence
- Content Word Count No Stopwords
- Noun Count
- Noun Incidence
- Verb Count
- Verb Incidence
- Adjective Count
- Adjective Incidence
- Adverb Count
- Adverb Incidence
- Number Count
- Number Incidence
- Type Count
- Type Token Ratio
- Comma Count
- Comma Incidence
- Average Word Length In Syllables
- Complex Word Count
- Complex Word Incidence,
- Average Sentence Length
- Negation Count
- Negation Incidence
- Negation In Stem
- NP Count
- NP Incidence
- Average NP Length
- NP Count With Embedding
- NP Incidence With Embedding
- Average All NP Length,
- PP Count
- PP Incidence
- PPs Per Sentence Ratio
- VP Count

- VP Incidence
- Passive Active Ratio
- Proportion Active VPs
- Proportion Passive VPs
- Agentless Passive Count
- Relative Clauses Count
- Relative Clauses Incidence
- Proportion Relative Clauses
- Polysemic Word Count
- Polysemic Word Incidence
- Average Sense No Content Words
- Average Sense No Nouns
- Average Sense No Verbs
- Average Sense No Non Auxiliary Verbs
- Average Sense No Adjectives
- Average Sense No Adverbs
- Average Noun Distance To WNRoot
- Average Verb Distance To WNRoot,
- Average Noun And Verb Distance To WNRoot
- Answer Words In Word Net Ratio
- Average Word Frequency Abs
- Average Word Frequency Rel
- Average Word Frequency Rank
- Average Content Frequency Abs
- Average Content Frequency Rel
- Average Content Frequency Rank
- Not In First 2000 Count
- Not In First 2000 Incidence
- Not In First 3000 Count
- Not In First 3000 Incidence
- Not In First 4000 Count
- Not In First 4000 Incidence
- Not In First 5000 Count
- Not In First 5000 Incidence
- Imagability
- Imagability Found Only
- Imagability Ratio
- Familiarity
- Familiarity Found Only
- Familiarity Ratio
- Concreteness
- Concreteness Found Only
- Concreteness Ratio
- Age Of Acquisition
- Age Of Acquisition Found Only
- Age Of Acquisition Ratio
- Meaningfulness Colorado Found Only
- Meaningfulness Pavio Found Only
- No Imagability Rating
- No Familiarity Rating
- No Concreteness Rating
- No Age of Acquisition Rating
- Connectives Count
- Connectives Incidence
- Additive Connectives Count
- Additive Connectives Incidence
- Temporal Connectives Count
- Temporal Connectives Incidence
- Causal Connectives Count
- Causal Connectives Incidence
- Referential Pronoun Count,
- Referential Pronoun Incidence

G Discrimination UMAP plots

In the following section, we provide the UMAP reduction plots for the discrimination parameters (darker being more discriminatory), as well as the prediction error UMAP plot for our best model (darker meaning higher error).

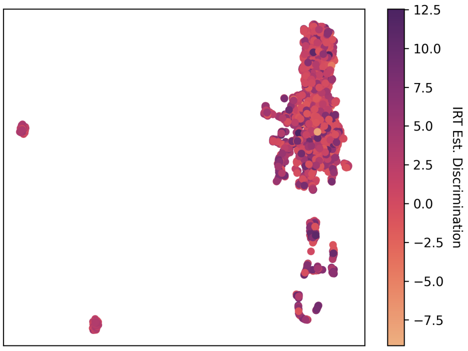


Figure 4: Answer BERT UMAP Reduction VS Discrimination values, train/dev set

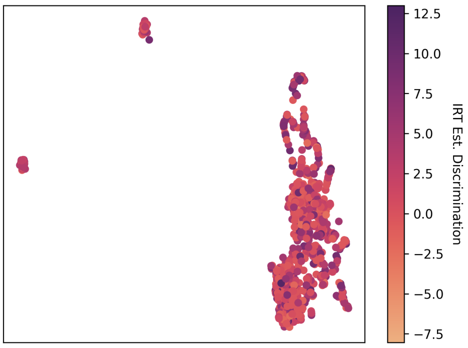


Figure 5: Answer BERT UMAP Reduction VS Discrimination values, test set

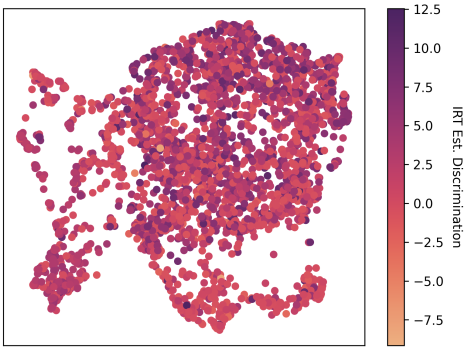


Figure 6: Question BERT UMAP Reduction VS Discrimination values, train/dev set

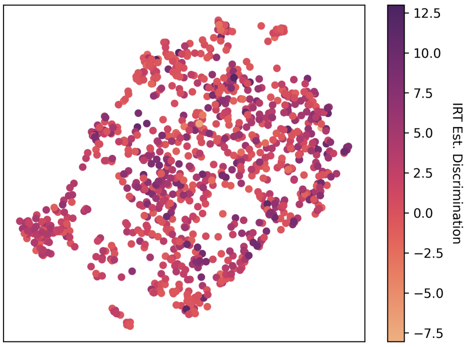


Figure 7: Question BERT UMAP Reduction VS Discrimination values, test set

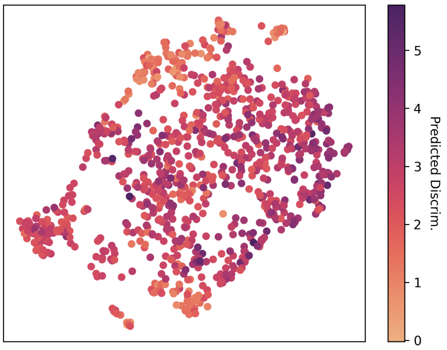


Figure 8: Question BERT UMAP Reduction VS Predicted Discrimination values, test set

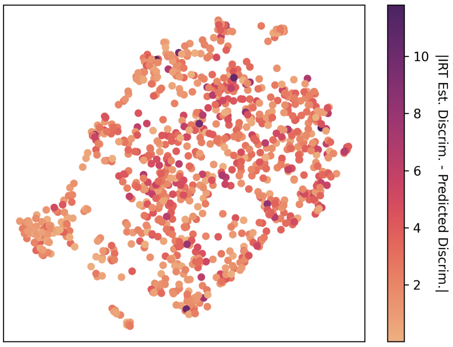


Figure 9: Question BERT UMAP Reduction VS Discrimination prediction error, test set

How does the pre-training objective affect what large language models learn about linguistic properties?

Ahmed Alajrami and **Nikolaos Aletras**
Department of Computer Science
University of Sheffield, UK
{ajsalajrami1, n.aletras}@sheffield.ac.uk

Abstract

Several pre-training objectives, such as masked language modeling (MLM), have been proposed to pre-train language models (e.g. BERT) with the aim of learning better language representations. However, to the best of our knowledge, no previous work so far has investigated how different pre-training objectives affect what BERT learns about linguistics properties. We hypothesize that linguistically motivated objectives such as MLM should help BERT to acquire better linguistic knowledge compared to other non-linguistically motivated objectives that are not intuitive or hard for humans to guess the association between the input and the label to be predicted. To this end, we pre-train BERT with two linguistically motivated objectives and three non-linguistically motivated ones. We then probe for linguistic characteristics encoded in the representation of the resulting models. We find strong evidence that there are only small differences in probing performance between the representations learned by the two different types of objectives. These surprising results question the dominant narrative of linguistically informed pre-training.[1]

1 Introduction

The most popular way to pre-train a transformer-based (Vaswani et al., 2017) language model (LM), e.g. BERT (Devlin et al., 2019), is by optimizing a masked language modeling (MLM) objective. The MLM task was inspired by the Cloze Task (Taylor, 1953), where humans were asked to guess omitted words in a sentence using its context, knowledge of syntax and other skills. The premise is that such an objective will guide a LM to encode linguistic information.

Apart from MLM, different types of objectives have been recently proposed. Yang et al. (2019) introduced a pre-training objective based on token order permutations. Clark et al. (2020) proposed a Replaced Token Detection pre-training task, that uses the output of a small MLM to corrupt the input by replacing some of the tokens. It then trains a discriminative model to predict if a token has been replaced or not. Aroca-Ouellette and Rudzicz (2020) explored various sentence and token-level auxiliary pre-training tasks (e.g. sentence ordering, term-frequency prediction), as better alternatives to the next sentence prediction (NSP) auxiliary task originally used to train BERT. Lan et al. (2020) introduced the sentence-order prediction task that focuses on the inter-sentence coherence, by predicting if two contiguous sentences have been swapped or not. Iter et al. (2020) proposed another inter-sentence pre-training task, that helps LMs to encode discourse relationships between sentences using contrastive learning. Yamaguchi et al. (2021) showed that a non-linguistically intuitive task (i.e. masked first character prediction) can effectively be used for pre-training.

Meanwhile, several studies have explored how well and to what extent LMs learn linguistic information. This is usually examined using probing tasks, i.e. simple classification tasks that test the LM's encodings for a single linguistic feature such as grammatical information. It has been found through probing that BERT encodes syntactic (Tenney et al., 2019; Liu et al., 2019; Miaschi and Dell'Orletta, 2020; Hewitt and Manning, 2019; Jawahar et al., 2019) and semantic information (Ettinger, 2020; Jawahar et al., 2019; Tenney et al., 2019). However, Hall Maudslay and Cotterell (2021) argue that BERT's syntactic abilities may have been overestimated.

In this paper, we hypothesize that linguistically motivated objectives (e.g. MLM) should help BERT to acquire better linguistic knowledge compared to using non-linguistically motivated objectives, i.e. tasks that are hard for humans to guess

[1] Code and models are available here: https://github.com/aajrami/acl2022-pre-training-objectives-probing

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 131 - 147
May 22-27, 2022 ©2022 Association for Computational Linguistics

the association between the input and the label to be predicted. To this end, we seek to answer the following research question: *How does the pre-training objective affect what LMs learn about the English language?*

Our findings challenge the MLM status quo, showing that pre-training with non-linguistically informative objectives (§2) results in models with comparable linguistic capabilities, as measured by standard probing benchmarks (§3). These surprising results (§4) suggest that careful analysis of how LMs learn is critical to further improve language modeling (§5).

2 Pre-training Objectives

We experiment with five different pre-training objectives. Two of them are considered linguistically motivated while the rest are not.

2.1 Linguistically Motivated Objectives

Masked Language Modeling (MLM): We use MLM as our first linguistically motivated pre-training objective. First introduced by Devlin et al. (2019), MLM randomly chooses 15% of the tokens from the input sentence and replaces 80% of them with a [MASK] token, 10% with a random token, and 10% remain unchanged.

Manipulated Word Detection (S+R): We also experiment with a simpler linguistically motivated objective, where the model selects and replaces 10% of input tokens with shuffled tokens from the same input sequence. Concurrently, it selects and replaces another 10% of input tokens with random tokens from the vocabulary (Yamaguchi et al., 2021).

2.2 Non-Linguistically Motivated Objectives

We assume that tasks that are hard for humans (such as a completely random prediction task) will make less likely the deeper layers of BERT (i.e. closer to the output layer) to acquire meaningful information about language. We also hypothesize that layers closer to the input might learn word co-occurrence information (Sinha et al., 2021).

Masked First Character Prediction (First Char): For our first non-linguistically motivated pre-training objective, we use the masked first character prediction introduced by Yamaguchi et al. (2021). In this task, the model predicts only the first character of the masked token (e.g. '[c]at' and '[c]omputer' belong to the same class). The model predicts the first character as one of 29 classes, including the English alphabet and digit, punctuation mark, and other character indicators.

Masked ASCII Codes Summation Prediction (ASCII): We also propose a new non-linguistically motivated pre-training objective, where the model has to predict the summation of the ASCII code values of the characters in a masked token. To make this harder and keep the number of classes relatively small, we define a 5-way classification task by taking the modulo 5 of the ASCII summation: $V = [\sum_i ascii(char_i)] \, \%5$. Guessing the association between the input and such label, is an almost impossible task for a human.

Masked Random Token Classification (Random): Finally, we propose a completely random objective where we mask 15% of the input tokens and we assign each masked token a class from 0 to 4 *randomly* for a 5-way classification similar to the ASCII task. We assume that a model pre-trained with a random objective should not be able to learn anything meaningful about linguistic information.

3 Probing Tasks

Probing tasks (Adi et al., 2016; Conneau et al., 2018; Hupkes et al., 2018) are used to explore in what extent linguistic properties are captured by LMs. A model is normally trained, using the representations of a language model, to predict a specific linguistic property. If it achieves high accuracy, it implies that the LM encodes that linguistic property. In this work, we use nine standard probing tasks introduced by Conneau et al. (2018) to examine the representation output for each layer of the different LMs we pre-train following Shen et al. (2020). These tasks probe for surface, syntactic and semantic information. The dataset for each probing task contains 100k sentences for training, 10k sentences for validation and another 10k sentences for testing.[2] We train a multi-layer perceptron (MLP) classifier for each probing task using the recommended hyperparameters in the SentEval toolkit (Conneau and Kiela, 2018).

Surface information task: **SentLen** aims for correctly predicting the number of words in a sentence.

[2] The datasets are all publicly available by Conneau and Kiela (2018).

Model	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	GLUE Avg.
BASE - 40 Epochs Pre-training (Upper Bound)									
MLM + NSP	83.8	90.8	87.8	69.9	91.9	85.0	58.9	89.3	82.1 (0.4)
BASE - 500k Steps Pre-training									
MLM	**81.4**	**89.0**	**86.5**	65.1	**90.6**	**86.0**	52.8	**87.2**	**79.8** ± 0.3
S+R	79.2	88.1	86.0	**67.7**	88.5	85.9	**55.8**	**87.2**	**79.8** ± 0.3
First Char	78.8	87.2	85.4	60.0	89.1	83.5	44.5	85.1	76.7 ± 0.4
ASCII	76.8	85.3	84.3	60.8	87.9	82.2	42.0	82.4	75.2 ± 0.3
Random	67.5	63.3	74.9	53.5	81.7	71.8	15.1	23.3	56.4 ± 0.4
MEDIUM - 250k Steps Pre-training									
MLM	**78.3**	85.6	85.2	62.2	**90.0**	82.0	44.3	84.0	**76.4** ± 0.4
S+R	76.2	85.5	84.8	**62.5**	86.5	79.8	**46.1**	**84.4**	75.7 ± 0.1
First Char	77.7	**85.7**	**85.4**	58.8	88.7	**82.6**	37.4	83.5	75.0 ± 0.3
ASCII	75.1	84.4	83.8	56.6	87.1	80.5	34.8	81.2	72.9 ± 0.4
Random	72.9	81.4	83.1	54.7	84.0	73.7	27.3	76.9	69.3 ± 0.5
SMALL - 250k Steps Pre-training									
MLM	**75.8**	**84.6**	84.4	**59.7**	**89.0**	**81.7**	**38.7**	**83.6**	**74.7** ± 0.4
S+R	75.1	84.2	84.4	55.8	85.6	76.0	36.6	82.5	72.5 ± 0.2
First Char	74.5	83.3	**84.5**	56.3	87.3	78.4	35.4	81.4	72.6 ± 0.4
ASCII	72.9	82.3	83.1	55.7	87.0	72.2	32.8	77.1	70.4 ± 0.2
Random	70.7	81.0	82.4	54.4	84.2	72.5	23.4	76.2	68.1 ± 0.6

Table 1: Results on GLUE dev sets with standard deviations over five runs. **Bold** values denote the best performance across each GLUE task and GLUE Avg. for each model setting.

Syntactic information tasks: **TreeDepth** tests if the representations preserve information about the hierarchical structure of a sentence, by predicting the depth of its parse tree. **TopConst** predicts the top constituents of the parse tree of a sentence. **BShift** tests if two adjacent words have been inverted or not.

Semantic information tasks: **Tense** aims to predict if the main-clause verb is present or past. **SubjNum** predicts if the subject of the main clause is singular or plural. **ObjNum** tests if the direct object of the main clause is singular or plural. Semantic Odd Man Out **(SOMO)** tests if a noun or verb has been replaced with another noun or verb. **CoordInv** predicts if a sentence made of two coordinate clauses has been inverted or not.

4 Experiments & Results

4.1 Experimental Setup

Models We pre-train BERT-BASE (Devlin et al., 2019) models by replacing MLM and the next sentence prediction (NSP) objectives, with one of the linguistically or non-linguistically motivated pre-training objectives (§2). For completeness, we also pre-train two smaller model architectures, MEDIUM and SMALL from (Turc et al., 2019) as in Yamaguchi et al. (2021). The MEDIUM model has eight hidden layers and eight attention heads. The SMALL model has four hidden layers and eight attention heads. Both, MEDIUM and SMALL, models have feed-forward layers of size 2048 and hidden layers of size 512. More details on hyperprameters can be found in Appendix A.

Pre-training Data All models are pre-trained on the BookCorpus (Zhu et al., 2015) and English Wikipedia from Hugging Face.[3] The text is tokenized using Byte-Pair-Encoding (Sennrich et al., 2016), resulting to a total of 2.7 billion tokens.

Pre-training Details Due to limited computational resources, each BASE model is pre-trained for 500k steps, while each MEDIUM and SMALL model is pre-trained for 250k steps using 8 NVIDIA Tesla V100 (SXM2 - 32GB). We use a batch size of 32 for BASE, and 64 for MEDIUM and SMALL. We optimize the models using Adam (Kingma and Ba, 2014).

Fine-tuning Details We use the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018) to fine-tune each model for up to 20 epochs with early stopping. For each fine-tuning task, we use five different seeds and

[3] https://github.com/huggingface/datasets

Model	SentLen (Surface)	TreeDepth (Syntactic)	TopConst (Syntactic)	BShift (Syntactic)	Tense (Semantic)	SubjNum (Semantic)	ObjNum (Semantic)	SOMO (Semantic)	CoordInv (Semantic)
	BASE - Jawahar et al. (2019)								
MLM+NSP	96.2	41.3	84.1	87.0	90.0	88.1	82.2	65.2	78.7
MLM+NSP (untrained)	92.5	29.8	55.2	50.1	63.8	67.4	63.7	50.6	50.3
	BASE - 500k Steps Pre-training								
MLM	**96.0** ± 0.2	41.5 ± 0.6	76.9 ± 0.2	86.5 ± 0.1	88.5 ± 0.7	87.4 ± 1.2	83.8 ± 0.2	**61.7** ± 0.5	65.5 ± 0.3
S+R	92.9 ± 0.4	**45.2** ± 0.6	**83.6** ± 0.2	**91.3** ± 0.7	87.8 ± 0.4	88.7 ± 0.2	84.5 ± 0.2	59.6 ± 0.4	**69.2** ± 0.3
First Char	93.7 ± 2.4	43.4 ± 1.2	81.1 ± 0.3	85.0 ± 0.4	86.0 ± 0.3	88.9 ± 0.1	**86.4** ± 0.1	56.5 ± 0.4	66.5 ± 0.8
ASCII	92.9 ± 0.4	43.3 ± 0.7	81.4 ± 0.4	82.7 ± 0.3	**88.7** ± 0.3	**89.1** ± 0.3	84.7 ± 0.5	54.0 ± 0.3	68.5 ± 0.8
Random	95.0 ± 0.6	39.6 ± 0.6	71.4 ± 1.0	68.9 ± 0.4	72.1 ± 0.5	74.3 ± 0.2	70.3 ± 0.1	50.4 ± 0.3	63.3 ± 0.3

Table 2: Mean accuracy with standard deviation over three runs for the best performing layer on the probing tasks using BASE models. **Bold** values denote the best performance across each probing task.

report the average. We report matched accuracy for MNLI task, Matthews correlation for CoLA task, Spearman correlation for STS-B task, accuracy for MRPC task, F1 scores for QQP task, and accuracy for all other tasks. The WNLI task is omitted following Aroca-Ouellette and Rudzicz (2020).

BERT Representations In all of the probing tasks, we use the BERT representations of the [CLS] token at every layer as the input to the probing classifier.

4.2 Fine-tuning Results

Table 1 shows the results of fine-tuning the models with all pre-training objectives on GLUE to measure their performance in downstream tasks. For the BASE model configuration, we observe that linguistically motivated objectives (e.g. MLM, S+R) achieve the best performance in downstream tasks. However, models pre-trained with non-linguistically motivated objectives (e.g. First Char, ASCII) still achieve competitive results. As expected, the model pre-trained using the Random objective obtains the lowest performance with 56.4 GLUE average score. However, its performance is still reasonable in many downstream tasks, suggesting that the model is able to learn some co-occurrence information from the input (Sinha et al., 2021; Yamaguchi et al., 2021). Similar behavior can be observed for the other two model configurations, MEDIUM and SMALL.

4.3 Probing Results

Table 2 presents the results of the best performing layer on the nine probing tasks using the representations from the BERT-BASE models as inputs to the MLP classifier. Similar to the fine-tuning results, we first observe that the predictive performance of models trained on representations learned using linguistically motivated objectives (e.g. MLM, S+R) achieve the best performance in six out of the nine probing tasks. However, *models trained on the representations learned using non-linguistically motivated objectives (e.g. First Char, ASCII) achieve very competitive results.*. For example, in the TopConst probing task, the model pre-trained using MLM pre-training objective achieves the best performance of 83.6%, while the the model pre-trained using ASCII pre-training objective achieves 81.4%.

Similar patterns can be observed from the probing results of the other two model configurations, MEDIUM and SMALL (see Tables 3 and 4 respectively). For instance, in the SentLen probing task in table 3, the difference between the best performing MEDIUM model (S+R) and the worst performing MEDIUM model (ASCII) is only 3.6%. In the ObjNum probing task in table 4, the SMALL model pre-trained using a non-linguistically motivated pre-training objective (ASCII) achieves 84.4%, while the SMALL models pre-trained using linguistically motivated pre-training objectives, MLM and S+R, achieve 83.5% and 83.3% respectively.

The full results of the probing tasks including all layers can be found in appendix B.

5 Discussion

Theoretically, LMs with non-linguistically motivated objectives would be expected to perform drastically worse than LMs pre-trained using MLM in both downstream tasks and linguistic capabilities. However, our results show that both types of LMs have surprisingly close performance (after fine-tuning on downstream tasks) and linguistic capabilities (after probing them) using the same training data, architecture and training scheme. We speculate that the pre-training data, and the size of

Model	SentLen	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
	(Surface)	(Syntactic)	(Syntactic)	(Syntactic)	(Semantic)	(Semantic)	(Semantic)	(Semantic)	(Semantic)
	MEDIUM - 250k Steps Pre-training								
MLM	92.3 ± 0.2	41.1 ± 0.1	76.9 ± 0.5	80.8 ± 0.1	85.9 ± 0.1	86.7 ± 0.1	83.7 ± 0.5	**56.1** ± 0.6	63.5 ± 0.7
S+R	**94.0** ± 0.5	**42.6** ± 0.2	**83.0** ± 0.5	**84.6** ± 0.3	85.7 ± 0.2	**87.9** ± 0.4	81.9 ± 0.5	55.8 ± 0.3	**66.5** ± 1.2
First Char	93.3 ± 0.3	40.4 ± 0.5	76.8 ± 0.3	80.3 ± 0.4	85.8 ± 0.5	86.3 ± 1.3	83.1 ± 0.1	53.8 ± 0.6	61.8 ± 0.3
ASCII	90.4 ± 0.5	40.5 ± 0.6	79.6 ± 0.2	80.0 ± 0.8	**87.8** ± 0.5	85.3 ± 0.3	83.9 ± 0.1	52.7 ± 0.4	64.7 ± 0.1
Random	92.9 ± 0.2	42.4 ± 0.8	71.5 ± 0.9	74.2 ± 0.0	86.1 ± 0.1	84.3 ± 0.3	**85.7** ± 0.3	51.3 ± 0.7	61.5 ± 0.4

Table 3: Mean accuracy with standard deviation over three runs for the best performing layer on the probing tasks using MEDIUM models. **Bold** values denote the best performance across each probing task.

Model	SentLen	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
	(Surface)	(Syntactic)	(Syntactic)	(Syntactic)	(Semantic)	(Semantic)	(Semantic)	(Semantic)	(Semantic)
	SMALL - 250k Steps Pre-training								
MLM	93.7 ± 0.4	41.6 ± 0.2	73.1 ± 0.2	78.3 ± 0.1	86.4 ± 0.7	83.5 ± 0.2	83.5 ± 0.1	**55.9** ± 0.6	**64.0** ± 0.3
S+R	**94.7** ± 0.8	**43.3** ± 1.0	76.8 ± 0.6	**82.1** ± 0.1	**86.5** ± 0.2	**85.6** ± 0.3	83.3 ± 0.5	54.9 ± 0.4	63.9 ± 0.1
First Char	90.7 ± 0.4	42.3 ± 0.4	**77.5** ± 0.1	76.2 ± 0.2	86.0 ± 0.1	84.7 ± 0.5	82.9 ± 0.7	52.4 ± 0.3	**64.0** ± 0.6
ASCII	89.9 ± 0.3	41.3 ± 0.4	74.6 ± 0.4	74.6 ± 0.1	85.7 ± 0.4	84.0 ± 0.3	**84.4** ± 0.2	52.3 ± 0.4	62.5 ± 0.1
Random	94.1 ± 1.0	42.6 ± 0.5	75.8 ± 0.4	71.0 ± 0.4	85.5 ± 0.5	83.8 ± 0.3	81.6 ± 0.3	50.7 ± 0.4	61.7 ± 0.5

Table 4: Mean accuracy with standard deviation over three runs for the best performing layer on the probing tasks using SMALL models. **Bold** values denote the best performance across each probing task.

the models have more impact on the effectiveness of LMs than the pre-training objectives. Furthermore, the comparable performance of different objectives in probing suggests that LMs mainly learn word co-occurrence information from pre-training (Sinha et al., 2021; Yamaguchi et al., 2021) and that the objectives may have a little effect to what actually learn about linguistic properties.

Recent studies have explored the limitations of using probing tasks to draw conclusions over a model's linguistic knowledge with some also suggesting improvements or alternative probing methods (Hewitt and Liang, 2019; Voita and Titov, 2020; Elazar et al., 2021; Maudslay and Cotterell, 2021). However, our results show no substantial differences in the performance across tasks that probe for syntactic or semantic information between models that have been pre-trained using linguistically motivated objectives or non-linguistically motivated ones.

6 Conclusions

In this work, we compared the linguistic capabilities of LMs. Surprisingly, our results show that pre-training with linguistically motivated objectives obtain comparable performance to non-linguistically motivated objectives. This suggests that the data and the size of the model could be more influential than the objectives themselves in language modeling. In future work, we plan to extend our experiments into other languages and probing tasks.

Acknowledgments

We would like to thank Katerina Margatina and George Chrysostomou for their invaluable feedback. We also thank the anonymous reviewers for their constructive feedback. AA is supported by the Centre for Doctoral Training in Speech and Language Technologies (SLT) and their Applications funded by UK Research and Innovation grant EP/S023062/1. NA is supported by EPSRC grant EP/V055712/1, part of the European Commission CHIST-ERA programme, call 2019 XAI: Explainable Machine Learning-based Artificial Intelligence.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*.

Stéphane Aroca-Ouellette and Frank Rudzicz. 2020. On Losses for Modern Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4970–4981, Online. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

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

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2021. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175.

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

Rowan Hall Maudslay and Ryan Cotterell. 2021. Do syntactic probes probe syntax? experiments with jabberwocky probing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 124–131, Online. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. *arXiv preprint arXiv:1909.03368*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138.

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. 2018. Visualisation and'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research*, 61:907–926.

Dan Iter, Kelvin Guu, Larry Lansing, and Dan Jurafsky. 2020. Pretraining with contrastive sentence objectives improves discourse performance of language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4859–4870.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019. Linguistic knowledge and transferability of contextual representations. *arXiv preprint arXiv:1903.08855*.

Rowan Hall Maudslay and Ryan Cotterell. 2021. Do syntactic probes probe syntax? experiments with jabberwocky probing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 124–131.

Alessio Miaschi and Felice Dell'Orletta. 2020. Contextual and non-contextual word embeddings: an in-depth linguistic investigation. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 110–119, Online. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Sheng Shen, Alexei Baevski, Ari S Morcos, Kurt Keutzer, Michael Auli, and Douwe Kiela. 2020. Reservoir transformers. *arXiv preprint arXiv:2012.15045*.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little.

In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wilson L Taylor. 1953. “cloze procedure”: A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. *arXiv preprint arXiv:1905.06316*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

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

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Atsuki Yamaguchi, George Chrysostomou, Katerina Margatina, and Nikolaos Aletras. 2021. Frustratingly simple pretraining alternatives to masked language modeling. *arXiv preprint arXiv:2109.01819*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.

Appendices

A Hyperparameter Details

We implement the models using PyTorch (Paszke et al., 2019) and the Transformers library (Wolf et al., 2020). We use maximum 10 epochs for BASE and MEDIUM, and 15 epochs for SMALL. We also use a learning rate of 1e-4 for MLM. 5e-5 for BASE First Char, S+R, and ASCII. 5e-6 for BASE Random. 1e-4 for SMALL and MEDIUM First Char, ASCII and Random. We also use weight decay of 0.01, attention dropout of 0.1, 10000 warmup steps. We also use 1e-8 Adam ϵ, 0.9 Adam β_1 and 0.999 Adam β_2.

B Results of each Probing Task

Tables 5 to 13 show the full results of each of the nine probing tasks for all model architectures and layers.

	SentLen				
Layer	**BASE - 500k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	95.4 ± 0.2	92.9 ± 0.4	90.7 ± 0.8	91.5 ± 0.3	92.6 ± 0.5
2	96.0 ± 0.2	92.9 ± 0.2	92.4 ± 0.4	91.7 ± 0.7	93.6 ± 0.3
3	95.3 ± 0.2	91.6 ± 0.6	92.9 ± 0.5	92.4 ± 1.7	94.4 ± 0.4
4	93.8 ± 1.2	92.2 ± 0.8	93.4 ± 1.3	92.9 ± 1.0	94.1 ± 0.6
5	93.9 ± 0.4	92.1 ± 0.6	93.7 ± 2.4	92.4 ± 0.5	93.8 ± 0.6
6	93.6 ± 0.5	92.4 ± 0.5	93.5 ± 1.7	92.1 ± 0.7	94.3 ± 0.4
7	92.6 ± 0.5	92.1 ± 0.8	93.1 ± 0.9	90.7 ± 1.4	94.4 ± 0.6
8	91.2 ± 0.5	91.7 ± 0.5	92.0 ± 1.6	89.9 ± 1.0	94.2 ± 1.0
9	89.0 ± 0.3	91.8 ± 0.4	90.9 ± 0.7	88.5 ± 1.6	95.0 ± 0.6
10	82.8 ± 0.7	91.1 ± 0.9	90.0 ± 0.9	86.7 ± 1.7	94.6 ± 0.1
11	79.4 ± 0.7	91.0 ± 0.4	88.6 ± 0.1	87.8 ± 0.5	94.4 ± 0.2
12	73.9 ± 0.3	90.1 ± 0.3	85.9 ± 0.1	86.4 ± 0.2	93.6 ± 0.4
Layer	**MEDIUM - 250k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	91.8 ± 0.5	88.4 ± 1.1	87.1 ± 0.8	86.6 ± 0.8	90.0 ± 0.9
2	92.3 ± 0.2	94.0 ± 0.5	93.3 ± 0.3	90.4 ± 0.5	92.3 ± 0.2
3	92.1 ± 0.2	94.0 ± 0.7	92.0 ± 0.6	89.2 ± 0.5	92.9 ± 0.2
4	91.7 ± 0.2	93.4 ± 0.7	91.4 ± 0.2	89.5 ± 0.5	92.2 ± 0.5
5	90.6 ± 0.3	92.7 ± 0.7	91.0 ± 0.2	89.7 ± 0.4	91.2 ± 0.7
6	89.3 ± 0.3	93.0 ± 0.6	90.1 ± 0.8	89.0 ± 0.5	88.7 ± 0.7
7	85.6 ± 0.2	92.0 ± 0.9	89.3 ± 0.5	86.1 ± 0.9	88.4 ± 0.7
8	70.5 ± 0.1	87.8 ± 1.4	84.9 ± 0.5	83.9 ± 0.5	83.2 ± 0.1
Layer	**SMALL - 250k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	92.9 ± 0.3	90.3 ± 1.3	89.8 ± 1.1	89.9 ± 0.3	94.1 ± 1.0
2	93.7 ± 0.4	93.8 ± 0.4	90.7 ± 0.4	88.7 ± 0.2	93.3 ± 1.1
3	91.7 ± 0.2	94.7 ± 0.8	89.7 ± 0.2	86.8 ± 0.5	90.1 ± 1.3
4	77.2 ± 0.3	93.0 ± 0.5	84.4 ± 0.5	85.5 ± 0.4	84.7 ± 0.3

Table 5: Results of the Sentence Length (SentLen) probing task for each layer of the pre-trained models.

			TreeDepth		
Layer			**BASE - 500k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	40.0 ± 0.6	36.6 ± 0.6	35.7 ± 0.2	36.1 ± 0.5	33.5 ± 0.7
2	41.2 ± 1.1	38.6 ± 0.9	37.7 ± 0.5	36.6 ± 0.3	35.9 ± 0.5
3	41.5 ± 0.6	40.0 ± 0.8	38.9 ± 0.6	37.1 ± 0.4	36.2 ± 0.4
4	40.3 ± 0.7	41.7 ± 0.6	39.4 ± 0.6	37.7 ± 0.9	36.9 ± 0.4
5	40.3 ± 1.1	44.2 ± 0.5	39.3 ± 0.3	38.4 ± 1.2	36.7 ± 0.5
6	40.9 ± 0.7	45.0 ± 0.3	40.6 ± 0.4	40.7 ± 0.5	36.5 ± 0.5
7	40.8 ± 0.8	44.9 ± 0.8	42.1 ± 0.6	42.4 ± 0.6	37.0 ± 0.6
8	40.0 ± 0.7	45.0 ± 0.7	43.4 ± 1.2	43.3 ± 0.7	39.0 ± 0.3
9	38.8 ± 1.1	44.3 ± 0.7	43.2 ± 1.3	43.3 ± 0.7	39.2 ± 0.3
10	37.4 ± 0.3	45.2 ± 0.6	43.4 ± 1.1	42.9 ± 0.5	39.3 ± 0.5
11	38.7 ± 0.6	44.5 ± 0.4	42.9 ± 1.2	42.7 ± 0.5	39.6 ± 0.6
12	38.3 ± 0.3	42.1 ± 0.7	41.5 ± 0.7	42.3 ± 0.4	37.9 ± 1.3
Layer			**MEDIUM - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	37.9 ± 0.2	37.8 ± 0.5	36.4 ± 0.3	37.4 ± 0.1	36.1 ± 0.5
2	39.0 ± 0.5	39.0 ± 1.2	36.5 ± 0.4	38.0 ± 0.4	36.4 ± 0.6
3	39.4 ± 0.2	40.4 ± 0.5	36.3 ± 0.2	37.7 ± 0.6	38.3 ± 0.6
4	40.5 ± 0.5	40.3 ± 0.6	36.7 ± 0.3	38.3 ± 0.3	41.6 ± 0.6
5	41.1 ± 0.1	41.8 ± 1.0	36.9 ± 0.6	39.1 ± 0.5	42.4 ± 0.8
6	40.5 ± 0.2	42.6 ± 0.2	37.5 ± 0.7	40.5 ± 0.6	40.5 ± 1.1
7	39.3 ± 0.2	42.5 ± 0.4	40.4 ± 0.5	39.1 ± 0.8	39.1 ± 0.5
8	38.6 ± 0.9	38.5 ± 0.6	40.2 ± 0.2	40.5 ± 0.1	35.6 ± 0.1
Layer			**SMALL - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	37.8 ± 0.3	39.2 ± 0.2	39.1 ± 0.3	37.5 ± 0.2	38.0 ± 0.2
2	40.1 ± 0.5	41.9 ± 0.6	40.6 ± 0.7	37.4 ± 0.2	41.6 ± 0.4
3	39.9 ± 0.9	41.6 ± 0.4	41.2 ± 0.3	41.3 ± 0.4	42.6 ± 0.5
4	41.6 ± 0.2	43.3 ± 1.0	42.3 ± 0.4	40.9 ± 0.6	39.2 ± 0.3

Table 6: Results of the Tree Depth (TreeDepth) probing task for each layer of the pre-trained models.

TopConst					
Layer	**BASE - 500k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	62.0 ± 0.3	70.2 ± 0.7	60.9 ± 0.4	66.7 ± 1.1	65.2 ± 0.2
2	72.6 ± 0.4	73.7 ± 0.2	69.3 ± 0.2	67.7 ± 0.2	68.4 ± 0.6
3	74.0 ± 0.5	79.6 ± 0.8	70.7 ± 0.5	69.2 ± 0.2	69.3 ± 0.1
4	73.0 ± 0.5	81.4 ± 0.4	71.0 ± 0.1	70.8 ± 0.3	69.9 ± 0.4
5	73.7 ± 0.5	83.6 ± 0.2	71.3 ± 0.3	70.6 ± 0.5	69.8 ± 1.1
6	74.6 ± 0.6	83.1 ± 0.7	71.7 ± 0.5	75.4 ± 0.9	69.2 ± 0.6
7	75.1 ± 0.7	82.4 ± 0.2	76.2 ± 0.5	78.4 ± 0.5	70.0 ± 1.1
8	76.9 ± 0.2	81.6 ± 0.4	78.2 ± 0.3	78.5 ± 0.4	71.4 ± 1.0
9	76.8 ± 0.4	81.7 ± 0.6	80.1 ± 0.3	80.4 ± 0.2	70.7 ± 0.6
10	74.6 ± 0.6	80.6 ± 0.7	81.1 ± 0.3	81.4 ± 0.4	71.2 ± 1.1
11	74.2 ± 0.1	79.6 ± 0.9	80.7 ± 0.4	81.3 ± 0.6	69.8 ± 0.6
12	72.5 ± 0.2	76.5 ± 0.5	79.9 ± 0.2	81.0 ± 0.2	67.4 ± 0.4
Layer	**MEDIUM - 250k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	64.9 ± 0.3	63.1 ± 1.7	67.6 ± 0.6	68.2 ± 0.6	55.3 ± 0.3
2	72.1 ± 0.6	69.8 ± 0.6	68.7 ± 0.5	70.5 ± 1.2	61.9 ± 1.0
3	72.1 ± 0.6	72.3 ± 0.8	68.3 ± 0.7	69.1 ± 1.0	66.0 ± 1.4
4	72.6 ± 0.6	80.6 ± 0.3	69.1 ± 0.6	74.2 ± 0.6	69.8 ± 0.4
5	74.8 ± 0.5	81.9 ± 0.6	69.8 ± 0.7	78.1 ± 0.7	71.5 ± 0.9
6	75.2 ± 0.4	81.9 ± 0.5	73.2 ± 0.1	79.3 ± 0.6	69.7 ± 0.8
7	76.9 ± 0.5	83.0 ± 0.5	75.7 ± 0.7	78.5 ± 0.5	70.7 ± 0.6
8	72.6 ± 0.3	79.8 ± 0.3	76.8 ± 0.3	79.6 ± 0.2	62.9 ± 0.2
Layer	**SMALL - 250k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	66.4 ± 0.2	69.2 ± 0.4	74.6 ± 0.3	66.3 ± 0.2	66.7 ± 1.4
2	72.5 ± 0.4	73.2 ± 0.2	75.8 ± 0.3	66.0 ± 0.5	74.2 ± 0.3
3	71.9 ± 0.3	73.8 ± 0.2	76.4 ± 0.6	72.6 ± 0.9	75.8 ± 0.4
4	73.1 ± 0.2	76.8 ± 0.6	77.5 ± 0.1	74.6 ± 0.4	72.7 ± 0.1

Table 7: Results of the Top Constituent (TopConst) probing task for each layer of the pre-trained models.

			BShift		
Layer			**BASE - 500k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0
2	50.0 ± 0.1	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0
3	56.6 ± 0.3	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0
4	57.9 ± 0.2	74.1 ± 0.3	50.0 ± 0.0	53.4 ± 0.4	50.0 ± 0.0
5	59.8 ± 0.1	80.7 ± 0.2	50.0 ± 0.0	50.8 ± 1.4	50.0 ± 0.0
6	60.0 ± 0.7	83.3 ± 0.4	50.0 ± 0.0	69.6 ± 1.4	50.0 ± 0.0
7	64.9 ± 0.8	85.6 ± 0.2	63.5 ± 0.6	73.7 ± 2.8	60.2 ± 1.7
8	72.0 ± 1.3	88.1 ± 0.1	74.4 ± 0.8	78.5 ± 1.5	66.9 ± 0.2
9	81.4 ± 0.7	89.5 ± 0.2	82.4 ± 0.7	81.7 ± 0.8	67.0 ± 0.3
10	85.6 ± 0.2	90.2 ± 0.3	84.8 ± 0.3	81.7 ± 1.4	68.4 ± 0.2
11	86.5 ± 0.1	91.2 ± 0.6	85.0 ± 0.4	82.7 ± 0.3	68.9 ± 0.4
12	82.3 ± 0.3	91.3 ± 0.7	83.3 ± 0.2	82.4 ± 0.2	68.4 ± 0.1
Layer			**MEDIUM - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0
2	49.8 ± 0.3	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0
3	49.6 ± 0.4	50.0 ± 0.0	50.0 ± 0.0	57.9 ± 0.5	65.6 ± 0.7
4	56.2 ± 0.7	64.9 ± 0.3	50.0 ± 0.0	58.1 ± 0.7	70.5 ± 0.4
5	64.9 ± 0.3	76.4 ± 0.4	50.9 ± 1.6	58.9 ± 0.8	74.2 ± 0.0
6	69.6 ± 0.7	79.6 ± 0.1	73.5 ± 1.3	67.9 ± 1.3	72.5 ± 1.5
7	80.8 ± 0.1	82.1 ± 0.3	79.9 ± 0.4	75.1 ± 2.7	73.7 ± 0.1
8	77.9 ± 0.5	84.6 ± 0.3	80.3 ± 0.4	80.0 ± 0.8	70.3 ± 0.6
Layer			**SMALL - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	50.0 ± 0.1	50.0 ± 0.0	50.4 ± 0.2	53.2 ± 0.8	50.7 ± 0.4
2	49.8 ± 0.2	61.9 ± 0.3	57.7 ± 0.1	60.2 ± 1.2	60.0 ± 0.6
3	60.8 ± 0.7	74.4 ± 0.0	65.3 ± 0.2	72.1 ± 0.6	68.7 ± 0.7
4	78.3 ± 0.1	82.1 ± 0.1	76.2 ± 0.2	74.6 ± 0.1	71.0 ± 0.4

Table 8: Results of the Bigram Shift (BShift) probing task for each layer of the pre-trained models.

			Tense		
Layer			**BASE - 500k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	79.5 ± 0.8	83.6 ± 0.1	81.3 ± 0.1	79.9 ± 0.8	67.9 ± 0.6
2	84.0 ± 0.7	84.3 ± 1.0	82.0 ± 0.2	80.3 ± 0.6	68.9 ± 0.8
3	83.3 ± 0.3	85.7 ± 0.7	82.7 ± 0.5	82.0 ± 0.8	69.1 ± 0.6
4	83.7 ± 0.7	86.3 ± 0.7	83.9 ± 0.7	82.9 ± 0.4	69.0 ± 0.3
5	85.0 ± 0.5	86.3 ± 0.7	84.3 ± 0.9	83.0 ± 0.4	68.8 ± 0.5
6	86.2 ± 0.2	87.8 ± 0.4	84.3 ± 0.9	85.3 ± 0.1	68.9 ± 0.4
7	87.0 ± 0.1	87.1 ± 0.8	84.7 ± 0.6	86.0 ± 0.5	69.1 ± 0.5
8	86.4 ± 0.8	87.2 ± 0.4	86.0 ± 0.3	86.1 ± 0.5	70.9 ± 0.1
9	85.8 ± 1.8	86.3 ± 0.0	85.9 ± 0.2	87.2 ± 0.2	71.4 ± 0.6
10	86.5 ± 1.5	85.9 ± 0.6	85.7 ± 0.8	88.5 ± 0.2	72.1 ± 0.5
11	88.5 ± 0.7	83.7 ± 0.8	86.0 ± 0.7	88.7 ± 0.3	72.1 ± 0.5
12	83.9 ± 0.0	81.7 ± 1.7	85.9 ± 0.5	88.6 ± 0.4	71.0 ± 0.4
Layer			**MEDIUM - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	85.1 ± 0.5	82.2 ± 0.6	83.2 ± 0.4	81.4 ± 0.2	79.6 ± 0.9
2	84.1 ± 0.5	84.0 ± 0.3	82.5 ± 0.3	82.4 ± 0.5	80.0 ± 0.8
3	84.8 ± 0.4	85.4 ± 0.3	82.7 ± 0.1	82.0 ± 0.5	82.6 ± 0.8
4	85.6 ± 0.6	85.5 ± 0.6	82.7 ± 0.4	83.4 ± 0.5	84.6 ± 0.7
5	85.9 ± 0.4	85.0 ± 0.4	83.7 ± 0.4	84.1 ± 0.8	86.1 ± 0.1
6	85.7 ± 0.8	85.7 ± 0.2	84.7 ± 0.7	85.4 ± 0.5	83.9 ± 1.5
7	85.9 ± 0.1	84.6 ± 0.5	85.8 ± 0.5	85.3 ± 0.5	84.9 ± 0.4
8	83.9 ± 0.5	82.8 ± 0.4	85.6 ± 0.5	87.8 ± 0.5	84.6 ± 0.5
Layer			**SMALL - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	86.3 ± 0.4	84.9 ± 0.2	84.7 ± 0.7	82.7 ± 0.6	84.4 ± 0.3
2	86.2 ± 0.6	85.6 ± 0.5	84.7 ± 0.8	82.9 ± 0.2	85.2 ± 0.5
3	86.4 ± 0.7	86.0 ± 0.2	84.7 ± 0.6	84.5 ± 0.8	85.5 ± 0.5
4	85.2 ± 0.6	86.5 ± 0.2	86.0 ± 0.1	85.7 ± 0.4	84.9 ± 0.3

Table 9: Results of the Tense (Tense) probing task for each layer of the pre-trained models.

			SubjNum		
Layer	**BASE - 500k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	75.1 ± 0.5	75.5 ± 0.3	75.7 ± 0.8	77.0 ± 0.1	69.5 ± 0.2
2	81.6 ± 0.3	80.2 ± 0.3	78.3 ± 0.3	78.0 ± 0.6	71.7 ± 0.4
3	82.3 ± 0.3	85.0 ± 0.1	79.1 ± 0.4	78.7 ± 0.5	72.4 ± 0.3
4	81.8 ± 0.3	86.2 ± 0.5	79.1 ± 0.6	79.5 ± 0.1	72.1 ± 0.5
5	83.0 ± 0.3	88.7 ± 0.2	80.3 ± 0.9	80.5 ± 0.2	72.8 ± 0.1
6	85.0 ± 0.2	88.2 ± 0.3	82.2 ± 0.5	84.1 ± 0.4	72.7 ± 0.5
7	84.9 ± 0.6	87.5 ± 0.5	84.3 ± 0.1	85.5 ± 0.4	73.4 ± 0.6
8	86.0 ± 0.3	87.0 ± 0.9	85.5 ± 0.2	86.9 ± 0.9	73.9 ± 0.7
9	87.2 ± 1.0	87.1 ± 0.3	87.9 ± 0.4	88.9 ± 0.6	73.7 ± 0.4
10	87.4 ± 1.2	86.5 ± 0.5	88.9 ± 0.1	89.1 ± 0.3	74.3 ± 0.2
11	86.2 ± 0.2	86.1 ± 0.4	88.1 ± 0.4	88.8 ± 0.3	74.1 ± 0.1
12	82.3 ± 0.2	84.3 ± 0.4	86.3 ± 0.4	88.2 ± 0.4	74.2 ± 0.3
Layer	**MEDIUM - 250k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	79.3 ± 0.7	77.3 ± 0.6	77.0 ± 0.3	77.2 ± 1.1	75.0 ± 1.1
2	80.7 ± 0.2	80.0 ± 0.1	78.2 ± 0.6	80.4 ± 0.5	79.9 ± 0.5
3	81.0 ± 0.4	83.0 ± 0.7	78.0 ± 0.5	79.6 ± 0.2	80.4 ± 0.5
4	82.5 ± 0.5	86.9 ± 0.3	79.3 ± 0.6	81.0 ± 0.9	83.4 ± 0.4
5	83.9 ± 0.3	87.9 ± 0.4	79.7 ± 0.4	82.5 ± 0.5	84.3 ± 0.3
6	84.5 ± 0.2	87.5 ± 0.3	83.4 ± 0.3	84.4 ± 0.3	83.1 ± 1.0
7	86.7 ± 0.1	87.3 ± 0.1	86.3 ± 1.3	85.1 ± 0.5	83.9 ± 0.2
8	82.5 ± 0.2	85.3 ± 0.5	85.7 ± 0.2	85.3 ± 0.3	81.0 ± 0.1
Layer	**SMALL - 250k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	78.0 ± 0.8	80.9 ± 0.3	81.2 ± 0.1	76.5 ± 0.4	79.3 ± 0.3
2	82.2 ± 0.2	82.5 ± 0.3	82.1 ± 0.4	76.5 ± 0.5	82.4 ± 0.6
3	83.5 ± 0.2	81.8 ± 1.1	82.6 ± 0.2	82.6 ± 0.3	83.8 ± 0.3
4	83.3 ± 0.4	85.6 ± 0.3	84.7 ± 0.5	84.0 ± 0.3	81.9 ± 0.1

Table 10: Results of the Subject Number (SubjNum) probing task for each layer of the pre-trained models.

	ObjNum				
Layer	**BASE - 500k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	75.6 ± 0.3	73.6 ± 0.3	76.5 ± 0.4	77.5 ± 0.3	64.9 ± 0.6
2	81.1 ± 0.1	77.0 ± 0.1	77.9 ± 0.9	77.7 ± 1.3	67.5 ± 0.4
3	80.5 ± 1.0	79.7 ± 0.5	78.5 ± 0.5	79.7 ± 0.7	68.0 ± 0.3
4	80.3 ± 0.8	81.9 ± 0.5	78.7 ± 3.0	78.6 ± 0.4	68.1 ± 0.1
5	80.4 ± 1.0	84.4 ± 1.1	79.2 ± 2.9	78.8 ± 1.1	68.4 ± 0.4
6	82.0 ± 0.1	84.5 ± 0.2	81.1 ± 1.3	82.2 ± 1.2	68.4 ± 0.6
7	82.1 ± 0.4	84.4 ± 0.1	84.0 ± 0.7	83.3 ± 0.8	69.2 ± 0.2
8	82.1 ± 1.0	84.0 ± 0.9	84.4 ± 0.8	84.3 ± 1.2	69.4 ± 0.2
9	82.9 ± 0.3	84.1 ± 0.5	86.4 ± 0.1	84.5 ± 1.4	69.7 ± 0.1
10	83.8 ± 0.2	82.9 ± 0.5	86.4 ± 0.2	84.7 ± 0.6	69.9 ± 0.2
11	83.3 ± 0.3	83.8 ± 0.3	86.0 ± 0.3	84.5 ± 0.2	70.3 ± 0.1
12	78.5 ± 0.3	81.1 ± 1.7	83.5 ± 0.2	84.7 ± 0.5	70.2 ± 0.3
Layer	**MEDIUM - 250k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	80.1 ± 0.3	76.2 ± 0.4	76.2 ± 0.6	76.0 ± 0.3	75.2 ± 0.1
2	80.1 ± 0.1	78.4 ± 0.2	77.8 ± 0.7	78.5 ± 0.6	76.4 ± 0.5
3	80.6 ± 0.0	80.9 ± 0.1	77.2 ± 0.0	77.7 ± 0.8	78.7 ± 0.3
4	80.7 ± 0.2	81.0 ± 0.4	78.1 ± 0.1	77.8 ± 1.0	84.6 ± 0.2
5	82.5 ± 0.3	81.2 ± 0.6	78.7 ± 0.5	81.5 ± 0.2	85.7 ± 0.3
6	82.9 ± 0.1	81.9 ± 0.5	81.1 ± 0.3	82.9 ± 0.4	84.2 ± 0.6
7	83.7 ± 0.5	80.8 ± 0.3	83.1 ± 0.1	82.6 ± 0.2	83.8 ± 0.0
8	80.2 ± 0.4	80.3 ± 0.5	81.8 ± 0.3	83.9 ± 0.1	82.2 ± 0.3
Layer	**SMALL - 250k Steps Pre-training**				
	MLM	S+R	First Char	ASCII	Random
1	78.2 ± 0.9	81.4 ± 0.2	77.8 ± 0.4	77.7 ± 0.4	78.2 ± 0.3
2	82.0 ± 0.2	82.4 ± 0.3	79.7 ± 0.2	78.5 ± 0.4	79.0 ± 0.4
3	83.5 ± 0.1	82.5 ± 0.4	80.4 ± 0.2	84.4 ± 0.2	81.6 ± 0.3
4	80.9 ± 0.2	83.3 ± 0.5	82.9 ± 0.7	83.8 ± 0.2	79.4 ± 0.1

Table 11: Results of the Object Number (ObjNum) probing task for each layer of the pre-trained models.

			SOMO		
Layer			**BASE - 500k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	50.0 ± 0.2	50.0 ± 0.2	50.0 ± 0.2	50.0 ± 0.2	50.0 ± 0.2
2	52.5 ± 0.7	51.6 ± 0.3	50.5 ± 1.1	50.0 ± 0.2	50.0 ± 0.2
3	54.4 ± 1.3	50.0 ± 0.2	51.8 ± 0.9	50.0 ± 0.2	50.0 ± 0.2
4	55.2 ± 0.5	53.7 ± 0.8	52.5 ± 0.5	50.7 ± 1.2	50.0 ± 0.2
5	55.8 ± 0.0	55.4 ± 0.1	52.1 ± 0.8	50.0 ± 0.2	50.0 ± 0.2
6	57.6 ± 0.7	56.1 ± 0.3	52.8 ± 0.2	50.0 ± 0.2	50.0 ± 0.2
7	58.2 ± 1.0	56.8 ± 0.5	52.8 ± 1.1	50.0 ± 0.2	50.0 ± 0.2
8	58.1 ± 0.6	56.9 ± 1.3	53.7 ± 0.7	50.0 ± 0.2	50.0 ± 0.2
9	59.1 ± 0.4	57.9 ± 1.5	54.1 ± 1.0	53.2 ± 0.9	50.0 ± 0.2
10	60.6 ± 0.5	58.5 ± 0.9	56.3 ± 0.7	53.4 ± 0.2	50.4 ± 0.3
11	61.7 ± 0.5	58.9 ± 0.6	56.5 ± 0.4	53.9 ± 1.0	50.2 ± 0.3
12	57.8 ± 0.4	59.6 ± 0.4	55.4 ± 1.0	54.0 ± 0.3	50.2 ± 0.5
Layer			**MEDIUM - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	51.6 ± 0.5	50.2 ± 0.3	50.0 ± 0.2	50.7 ± 0.8	50.0 ± 0.2
2	52.3 ± 0.7	51.1 ± 0.1	50.0 ± 0.2	52.2 ± 0.4	50.0 ± 0.2
3	53.2 ± 0.1	52.6 ± 0.4	50.0 ± 0.2	52.1 ± 0.3	50.0 ± 0.2
4	53.1 ± 0.8	52.9 ± 0.7	50.0 ± 0.2	51.3 ± 0.3	50.8 ± 0.3
5	53.5 ± 0.6	53.8 ± 0.6	50.0 ± 0.2	51.2 ± 0.4	51.0 ± 0.4
6	54.6 ± 1.1	53.9 ± 0.7	51.5 ± 1.5	51.3 ± 0.2	50.0 ± 0.1
7	56.1 ± 0.6	55.2 ± 0.6	53.2 ± 0.2	52.0 ± 0.2	51.3 ± 0.7
8	54.1 ± 0.1	55.8 ± 0.3	53.8 ± 0.6	52.7 ± 0.4	50.6 ± 0.3
Layer			**SMALL - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	52.5 ± 0.2	52.2 ± 0.2	51.8 ± 0.2	51.3 ± 0.3	50.4 ± 0.3
2	55.4 ± 0.2	54.3 ± 0.4	51.5 ± 0.2	51.1 ± 0.2	50.7 ± 0.4
3	55.9 ± 0.6	54.8 ± 0.8	51.5 ± 0.2	52.2 ± 0.0	50.6 ± 0.2
4	53.9 ± 0.7	54.9 ± 0.4	52.4 ± 0.3	52.3 ± 0.4	50.2 ± 0.5

Table 12: Results of the Semantic Odd Man Out (SOMO) probing task for each layer of the pre-trained models.

			CoordInv		
Layer			**BASE - 500k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	57.3 ± 1.1	56.5 ± 1.0	55.1 ± 1.6	50.0 ± 0.0	50.0 ± 0.0
2	61.0 ± 0.5	59.7 ± 0.5	58.0 ± 0.6	50.0 ± 0.0	51.7 ± 3.0
3	61.8 ± 0.8	63.5 ± 0.8	58.9 ± 0.3	57.2 ± 0.6	57.8 ± 0.3
4	61.2 ± 0.5	64.8 ± 1.4	59.4 ± 0.6	59.6 ± 0.5	52.3 ± 4.0
5	62.0 ± 0.6	67.6 ± 0.4	60.2 ± 0.7	59.1 ± 0.3	55.2 ± 4.5
6	62.8 ± 0.4	69.2 ± 0.3	59.6 ± 0.7	59.8 ± 1.6	58.2 ± 0.4
7	61.6 ± 0.6	68.0 ± 0.3	61.3 ± 0.9	61.5 ± 2.0	59.8 ± 0.2
8	62.1 ± 0.4	67.4 ± 0.4	63.4 ± 0.7	62.9 ± 2.1	61.4 ± 0.2
9	62.1 ± 1.0	66.9 ± 0.2	63.9 ± 0.9	66.0 ± 1.0	62.6 ± 1.0
10	64.4 ± 0.5	67.8 ± 0.2	65.6 ± 0.6	67.6 ± 1.1	63.0 ± 0.2
11	65.5 ± 0.3	67.7 ± 0.5	66.5 ± 0.8	68.4 ± 0.5	63.3 ± 0.3
12	63.7 ± 1.3	65.4 ± 0.4	64.4 ± 0.9	68.5 ± 0.8	61.3 ± 0.7
Layer			**MEDIUM - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	59.4 ± 0.2	57.7 ± 0.3	56.9 ± 0.8	56.7 ± 0.7	55.9 ± 1.6
2	63.5 ± 0.7	60.7 ± 0.8	56.7 ± 0.4	60.4 ± 0.5	57.9 ± 0.2
3	62.1 ± 0.0	63.6 ± 0.4	56.5 ± 0.1	59.3 ± 0.7	58.6 ± 0.2
4	62.5 ± 0.2	65.6 ± 1.0	56.0 ± 0.7	60.0 ± 0.7	61.5 ± 0.4
5	63.1 ± 0.3	66.2 ± 1.2	57.6 ± 1.2	60.2 ± 0.4	61.4 ± 0.5
6	62.5 ± 0.3	65.7 ± 1.5	58.3 ± 0.4	60.2 ± 1.0	60.1 ± 0.7
7	61.7 ± 0.6	66.5 ± 1.2	60.4 ± 0.9	60.1 ± 1.5	60.3 ± 0.7
8	58.4 ± 0.5	63.8 ± 1.8	61.8 ± 0.3	64.7 ± 0.1	58.7 ± 0.4
Layer			**SMALL - 250k Steps Pre-training**		
	MLM	S+R	First Char	ASCII	Random
1	61.4 ± 0.1	60.1 ± 0.6	62.8 ± 0.1	59.7 ± 0.4	58.9 ± 0.5
2	64.0 ± 0.3	62.2 ± 0.4	64.0 ± 0.6	59.1 ± 0.2	61.0 ± 0.8
3	62.2 ± 0.3	62.7 ± 0.3	63.0 ± 0.4	61.4 ± 0.2	61.7 ± 0.5
4	59.4 ± 0.5	63.9 ± 0.1	62.2 ± 0.3	62.5 ± 0.1	59.9 ± 0.2

Table 13: Results of the Coordination Inversion (CoordInv) probing task for each layer of the pre-trained models.

The Power of Prompt Tuning for Low-Resource Semantic Parsing

Nathan Schucher[1,2] **Siva Reddy**[2,3] **Harm de Vries**[1]
[1]ServiceNow Research
[2]Mila/McGill University
[3]Facebook CIFAR AI Chair
{nathan.schucher,harm.devries}@servicenow.com

Abstract

Prompt tuning has recently emerged as an effective method for adapting pre-trained language models to a number of language understanding and generation tasks. In this paper, we investigate prompt tuning for semantic parsing—the task of mapping natural language utterances onto formal meaning representations. On the low-resource splits of Overnight and TOPv2, we find that a prompt tuned T5-xl significantly outperforms its fine-tuned counterpart, as well as strong GPT-3 and BART baselines. We also conduct ablation studies across different model scales and target representations, finding that, with increasing model scale, prompt tuned T5 models improve at generating target representations that are far from the pre-training distribution.

1 Introduction

With the widespread success of pre-trained language models (LMs; Devlin et al. 2019; Raffel et al. 2020; Bommasani et al. 2021), it becomes increasingly important to explore how such models can be adapted to downstream tasks. One adaptation method which has recently attracted much attention is prompt design (Brown et al., 2020; Shin et al., 2020), which modulates the behaviour of a LM through a task description and a few input-output examples. Brown et al. (2020) show that this adaptation strategy is increasingly effective for larger LMs. However, prompt design is sensitive to the exact phrasing of the prompt, and, more importantly, performs worse than fine-tuning models on task-specific examples (Lester et al., 2021).

Prompt tuning has recently arisen as a strong performing alternative adaption method (Lester et al., 2021). Rather than hand-designing discrete prompts, prompt tuning optimizes the embeddings of a number of task-specific prompt tokens. In contrast to fine-tuning, this method keeps almost all LM parameters frozen. On a set of language understanding tasks, Lester et al. (2021) show that prompt tuning becomes competitive with fine-tuning for the largest pre-trained T5 models (Raffel et al., 2020). Li and Liang (2021) also explore a related parameter-efficient adaptation method called prefix-tuning, finding that it outperforms fine-tuning on low-resource natural language generation tasks.

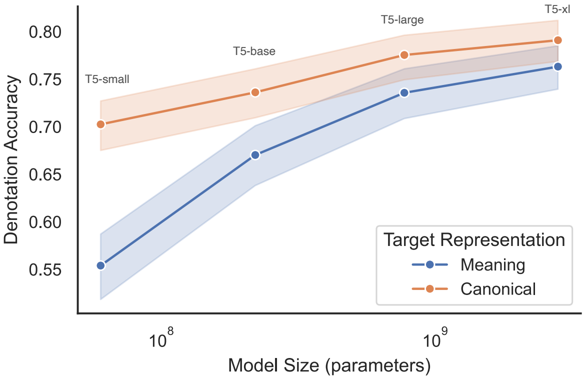


Figure 1: We show that the T5 prompt tuning performance difference between target representations shrinks as the number of parameters increase, with constrained decoded T5-xl achieving close to performance parity.

In this paper, we investigate prompt tuning for semantic parsing. This task is fundamentally different from the aforementioned language understanding and generation tasks, as it requires that models output formal meaning representations which do not resemble the natural language distribution seen during pre-training. In particular, we focus on the low-resource setup because examples for semantic parsing are difficult and expensive to collect (Wang et al., 2015; Marzoev et al., 2020). We therefore evaluate prompt tuning on two datasets: the 200-shot version of Overnight (Wang et al., 2015; Shin et al., 2021) and the low-resource splits TOPv2 (Chen et al., 2020). On both datasets, we compare prompt tuning T5 against fine-tuning and investigate the effect of canonicalizing the meaning

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 148 - 156
May 22-27, 2022 ©2022 Association for Computational Linguistics

representation, i.e. to what extent naturalizing the logical forms influences performance. In addition, we study the effect of T5 model scale on Overnight as well as varying data regimes on TOPv2. Our main findings can be summarized as follows:

- For large T5 models, prompt tuning significantly outperforms fine-tuning in the low-data regime, resulting in an absolute improvement of 6% and 15% on Overnight and TOPv2, respectively. This performance gap decreases when more training data becomes available.

- With growing model size, prompt tuned T5 models are increasingly capable of outputting diverse target representations (see Figure 1). On Overnight, we find that the disparity between canonical and meaning representations shrinks from 17% to 4% for T5-small and T5-xl, respectively. On TOPv2, prompt tuned T5-large models are much better at generating out-of-vocabulary tokens than T5-small.

2 Related work

Our work is related to recent work on semantic parsing and prompt tuning, which we briefly describe below.

2.1 Semantic Parsing

Semantic parsing is the task of converting a natural language utterance $\mathbf{u} = (u_1, \ldots, u_N)$ to a formal meaning representation $\mathbf{z} = (z_1, \ldots, z_M)$. These meaning representations, also referred to as logical forms, can be interpreted by machines and executed in a real environment. For example, ThingTalk (Campagna et al., 2019) and TOP (Gupta et al., 2018) are meaning representations for executing commands of virtual assistants, while SQL is a representation for interacting with relational databases. In recent years, neural sequence-to-sequence models have become the dominant approach for semantic parsing tasks (Dong and Lapata, 2016).

Canonicalization A common simplification step in semantic parsing is to canonicalize the meaning representations. That is, the meaning representation $\mathbf{z}$ is naturalized to a canonical form $\mathbf{c}$ through a grammar or set of rules. Examples of the meaning and canonical representation for Overnight and TOPv2 (Wang et al., 2015; Chen et al., 2020) can be found in Fig. 2.

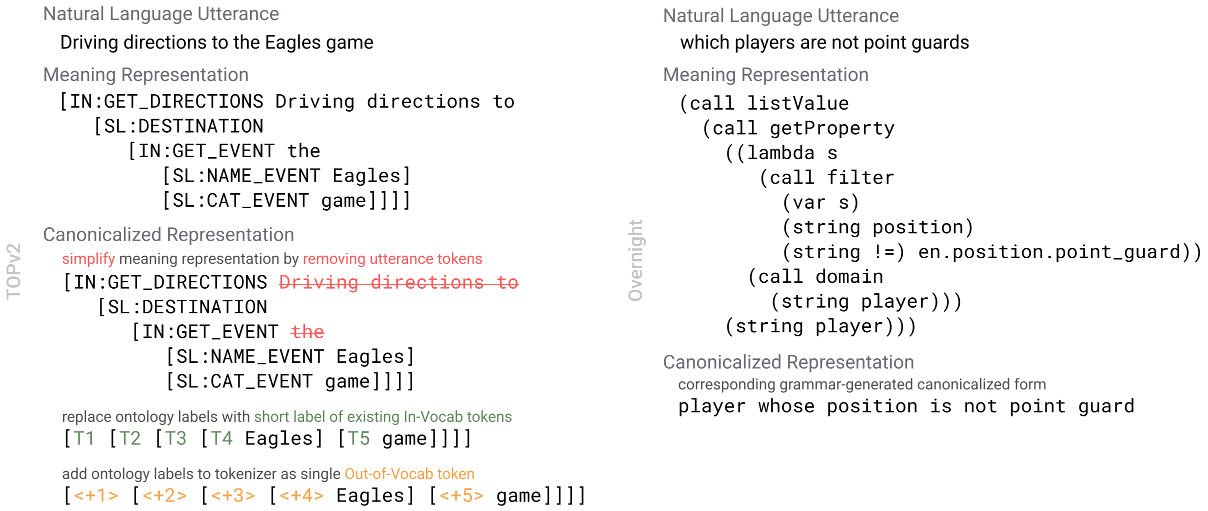
When canonical representations are available, Berant and Liang (2014) argue that semantic parsing can be seen as a paraphrase task. They propose to use a paraphrase model—using e.g. word vectors trained on Wikipedia—to find the best paraphrase of utterance $\mathbf{u}$ among a set of canonical utterances. They show this paraphrase model improves results over directly generating logical forms on two question-answering datasets. Marzoev et al. (2020) extends this work by showing that pre-trained language models like BERT can be effective paraphrasers. While Berant and Liang (2014); Marzoev et al. (2020) use models to score canonical utterances, Shin et al. (2021) propose to constrain the generation process of autoregressive models like BART and GPT-3. On a number of few-shot semantic parsing tasks, they demonstrate the benefit of generating canonical representations over meaning representations.

2.2 Prompt-tuning

Lester et al. (2021) evaluates prompt tuning on SuperGLUE, a benchmark consisting of eight language understanding tasks. They find that prompt tuning becomes competitive with fine-tuning for the largest T5 model. Li and Liang (2021) propose prefix-tuning to adapt BART and GPT-2 for natural language generation tasks. This method differs from Lester et al. (2021) in that it prepends trainable embeddings for each layer of the language model rather than introducing token embeddings at the input layer. They demonstrate that pre-fix outperforms fine-tuning baselines. Similarly, Liu et al. (2021) also show encouraging results for prompt tuning on natural language understand and generation tasks. Qin and Eisner (2021) also explores prompt tuning but for a knowledge extraction task. Inserting general adapter layers into pre-trained language models is also proposed in Houlsby et al. (2019); Mahabadi et al. (2021). Related to our work are also other few-shot adaptation techniques like PET (Schick and Schütze, 2021). Moreover, adapter layers have also been explored in the computer vision domain (Rebuffi et al., 2017; de Vries et al., 2017).

3 Experiments

To evaluate low-resource prompt tuning, we compare against fine-tuned variants of the same model on two semantic parsing datasets with canonical representations available. We compare both large



Driving directions to the Eagles game

```
[IN:GET_DIRECTIONS Driving directions to
   [SL:DESTINATION
      [IN:GET_EVENT the
         [SL:NAME_EVENT Eagles]
         [SL:CAT_EVENT game]]]]
```

simplify meaning representation by removing utterance tokens

[IN:GET_DIRECTIONS ~~Driving directions to~~
   [SL:DESTINATION
      [IN:GET_EVENT ~~the~~
         [SL:NAME_EVENT Eagles]
         [SL:CAT_EVENT game]]]]

replace ontology labels with short label of existing In-Vocab tokens

```
[T1 [T2 [T3 [T4 Eagles] [T5 game]]]]
```

add ontology labels to tokenizer as single Out-of-Vocab token

```
[<+1> [<+2> [<+3> [<+4> Eagles] [<+5> game]]]]
```

which players are not point guards

```
(call listValue
  (call getProperty
    ((lambda s
       (call filter
         (var s)
         (string position)
         (string !=) en.position.point_guard))
      (call domain
        (string player)))
    (string player)))
```

corresponding grammar-generated canonicalized form

```
player whose position is not point guard
```

Figure 2: Examples from the TOPv2 and Overnight datasets with the corresponding canonicalization schemes.

and small variants of the T5 architecture on these datasets and experiment with various canonicalized representations.

3.1 Datasets

Overnight The Overnight semantic parsing dataset (Wang et al., 2015) consists of 13,682 natural utterance, canonical form, meaning representation triples split across eight domains. To simulate low-resource splits of this dataset, we follow Shin et al. and create randomly subsampled splits of 200 training examples for each domain, using 20% of the remaining data for validation. We measure and report denotation accuracy by evaluating all predicted queries using the SEMPRE toolkit (Berant et al., 2013). We repeat each experiment on Overnight with five different random splits.

TOPv2 Chen et al. (2020) introduce the TOPv2 dataset, a task-oriented semantic parsing dataset with eight domains, two of which come with predefined low-resource splits. The authors propose a principled way of constructing low-resource training sets, *samples per intent and slot* (SPIS), intended to ensure equal exposure to ontology labels across domains of varying complexity. We experiment with the *weather* and *reminder* domains at the 10, 25, and 500 SPIS resource splits, performing five runs on each model varying the random seed. The *reminder* domain is the most challenging with 19 intent labels, 32 slot labels, and with 21% of the programs having a depth greater than 2. *Weather* in comparison has 7 intent labels, 11 slot labels, and no programs with depth greater than 2.

3.2 Canonicalized Representations

3.2.1 Overnight

Overnight uses a context-free synchronous grammar to generate canonical representations for the logical forms. As can be seen in Fig. 2, these canonical representations resemble natural language.

3.2.2 TOPv2

Chen et al. apply a set of simple modifications to the TOPv2 meaning representations to arrive at a canonical form used in all their experiments. Unlike Overnight, these pre-processing steps are largely small encoding differences and do not change the syntactic structure of the logical forms. We adopt all of these canonicalization steps (except for lexicographic sorting of the semantic parse tree) and add an ontology label shortening step. Examples of these transformations can be seen in Fig. 2 and are briefly described below.

Simplify removes redundant utterance tokens unnecessary for interpreting the meaning representation.

Out-of-Vocab adds the entire intent or slot label to the tokenizer as a new single tokens with a corresponding randomly initialized embedding.

In-Vocab replaces the intent and slot labels with a short unique identifier representable by the pre-trained tokenizer.

We perform an ablation over these canonicalization choices, repeating each experiment three times with varying random seed.

Model	Representation	Method	Bask.	Blo.	Cal.	Hou.	Pub.	Rec.	Res.	Soc.	Avg
T5-small	Meaning	FT	0.767	0.454	0.685	0.608	0.640	0.698	0.691	0.581	0.641
		PT	0.621	0.312	0.470	0.352	0.478	0.506	0.608	0.352	0.463
	Canonical	FT	0.775	0.466	0.721	0.616	0.665	0.673	0.636	0.568	0.640
		PT	0.764	0.440	0.680	0.601	0.648	0.699	0.697	0.578	0.638
T5-base	Meaning	FT	0.769	0.455	0.717	0.612	0.670	0.713	0.714	0.587	0.655
		PT	0.717	0.429	0.677	0.510	0.596	0.639	0.705	0.492	0.596
	Canonical	FT	0.800	0.466	0.736	0.642	0.711	0.694	0.696	0.597	0.668
		PT	0.786	0.452	0.682	0.636	0.675	0.705	0.733	0.614	0.660
BART	Meaning	FT	0.734	0.370	0.514	0.540	0.514	0.477	0.417	0.424	0.499
	Canonical	FT	0.591	0.331	0.740	0.309	0.668	0.598	0.582	0.532	0.544
T5-large	Meaning	FT	0.777	0.432	0.690	0.639	0.709	0.729	0.723	0.590	0.661
		PT	0.792	0.469	0.739	0.676	0.696	0.734	0.778	0.600	0.685
	Canonical	FT	0.793	0.458	0.760	0.658	0.678	0.727	0.715	0.581	0.671
		PT	0.819	0.525	0.768	0.712	0.744	0.789	0.769	0.655	0.723
T5-xl	Meaning	FT	0.774	0.413	0.702	0.630	0.682	0.691	0.705	0.580	0.647
		PT	0.819	0.532	0.767	0.693	0.694	0.758	0.778	0.632	0.709
	Canonical	FT	0.799	0.486	**0.781**	0.647	0.724	0.732	0.725	0.619	0.689
		PT	**0.839**	**0.544**	0.777	**0.729**	**0.770**	**0.791**	**0.789**	**0.702**	**0.743**

Table 1: Unconstrained denotation accuracy for all models (with unconstrained decoding) on the Overnight dataset. For each domain, we report the average over 5 runs trained on randomly sampled splits of 200 examples for fine-tuned (FT) and prompt tuned (PT) models.

3.3 Models

We provide training details and hyperparameters for all models in Appendix A. Below, we briefly explain the prompt-tuning methodology.

3.3.1 Prompt Tuning

Prompt tuning, as proposed by Lester et al. (2021), prepends a sequence of continuous embeddings $\mathbf{p} = (p_1, \ldots, p_K)$ to the sequence input embeddings $e(\mathbf{u}) = (e(u_1), \ldots, e(u_N))$ before feeding it to a language model with parameters θ. During prompt tuning we optimize the prompt embeddings $(p_1, \ldots, p_K)$ exclusively, keeping the language model parameters θ and the pretrained vocabulary embeddings fixed. Note that this process still requires backpropagating gradients through the full language model. Like fine-tuning models, we maximize the likelihood of generating the output sequence $\mathbf{z}$.

4 Results

In Table 1, we report Overnight results across four T5 model scales and two target representations. In Table 2, we add constrained decoding (see Appendix A) to our best performing T5 model and compare against previously reported Overnight results. In Table 3, we display the results of T5-large on the three different SPIS-splits of TOPv2, and include the BART-CopyPtr results from Chen et al. (2020). In Table 4, we summarize the results of the canonicalization ablation study for TOPv2.

4.1 Prompt tuning vs fine tuning

We find that prompt tuning improves over fine-tuning for all large model configurations and target representations. On Overnight, prompt tuned denotation accuracy exceeds fine-tuned counterparts by up to 5 points with T5-large and T5-xl. For T5-small and T5-base, prompt tuning remains competitive (within 1% average accuracy) with fine-tuning when predicting canonical forms. On TOPv2, prompt tuning achieves an absolute improvement of 15% mean accuracy over fine-tuning on the lowest SPIS split. This performance disparity lessens when training data increases; however, prompt tuned T5-large continues to beat its fine-tuned counterpart by 5 points at 500 SPIS and the BART-CopyPtr model by 1.4 points.

Our prompt tuning models outperform previously reported results on these datasets. On Overnight, our best model—T5-xl PT with canonical representations and constrained decoding—outperforms the BART FT model of Shin et al. (2021) by 5 accuracy points, and GPT-3 by more than 2 points. On the 25 SPIS split of TOPv2, we see an average improvement of more than 5 points compared to the BART-CopyPTR of Chen et al. (2020).

4.2 Canonical vs meaning representations

Our main finding is that prompt tuned T5 models become better at generating meaning representations with increased model size. On Overnight, we see the absolute difference between canonical and meaning representations shrink from 17.5 points

Model	Representation	Method	Decoding	Bask.	Blo.	Cal.	Hou.	Pub.	Rec.	Res.	Soc.	Avg
T5-xl	Meaning	PT	Constrained	0.841	0.592	0.802	0.765	0.776	0.814	0.789	0.725	0.763
T5-xl	Canonical	PT	Constrained	0.856	0.619	0.806	**0.779**	**0.824**	0.830	0.822	**0.793**	**0.791**
BART†	Meaning	FT	Constrained	0.834	0.499	0.750	0.619	0.739	0.796	0.774	0.620	0.704
BART†	Canonical	FT	Constrained	**0.864**	0.554	0.780	0.672	0.758	0.801	0.801	0.666	0.737
GPT-2†	Meaning	FT	Constrained	0.760	0.479	0.736	0.571	0.645	0.699	0.660	0.606	0.644
GPT-2†	Canonical	FT	Constrained	0.836	0.540	0.766	0.666	0.715	0.764	0.768	0.623	0.710
GPT-3†	Canonical	Context	Constrained	0.859	**0.634**	0.792	0.741	0.776	0.792	0.840	0.687	0.765
GPT-3†*	Meaning	Context	Constrained	0.680	0.530	0.680	0.580	0.630	0.750	0.780	0.630	0.657
GPT-3†*	Canonical	Context	Constrained	0.800	0.620	**0.820**	0.710	0.790	**0.840**	**0.890**	0.720	0.774

Table 2: Constrained denotation accuracy for all models on the Overnight dataset. For each domain, we report the average over 5 runs trained on randomly sampled splits of 200 examples. † denotes results reported by Shin et al. (2021). * indicates performance on subsampled test set.

SPIS	Model	Method	Reminder	Weather	Average
10	T5-large	FT	0.392	0.579	0.486
		PT	**0.567**	**0.700**	**0.634**
25	BART-CopyPtr	FT	0.557	0.716	0.637
	T5-large	FT	0.502	0.683	0.593
		PT	**0.642**	**0.739**	**0.691**
500	BART-CopyPtr	FT	0.719	**0.849**	0.784
	T5-large	FT	0.649	0.846	0.748
		PT	**0.749**	0.847	**0.798**

Table 3: Average exact match accuracies (5 runs) for different low-resource splits of the TOPv2 dataset. BART-CopyPtr results from Chen et al. (2020).

	None		Simplified		In-Vocab		Out-of-Vocab	
SPIS	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.
10	0.43	**0.70**	0.31	0.66	0.45	0.64	0.23	0.69
25	0.56	**0.74**	0.51	0.73	0.55	0.71	0.27	0.70
500	0.72	**0.85**	0.72	**0.85**	0.72	**0.85**	0.48	0.83

Table 4: Exact match accuracies (3 runs) on TOPv2 Weather domain for different meaning representation canonicalization choices (**bold** indicates best exact match accuracy at that resource level), Sm. and Lg. refer to T5-small and T5-large, respectively.

for T5-small to 3.4 points for T5-xl (Table 1). This gap shrinks another 18% to 2.8 points when we apply constrained decoding to T5-xl (Table 2). By contrast, Shin et al. (2021) reports an 11.7 point difference when prompting GPT-3. For our fine-tuning baselines, we observe a small performance gap of 4 points across target representations for BART and T5-xl, while we observe no gap for T5-small, T5-base, and T5-large models.

In our TOPv2 experiments we find similar evidence of large T5 model flexibility for generating sequences far from the training distribution. In particular, for our most intrusive canonicalization scheme `Out-of-Vocab`, which adds novel tokens to the vocabulary and leaves these embeddings un-trained, we find no significant reduction in performance for T5-large across all data resource levels. T5-small, in comparison, sees almost a 50% drop in performance relative to no canonicalization (`None`) at the 10 SPIS level and continues to underperform by 33 % at the 500 SPIS level.

Interestingly, we find that `In-Vocab` drastically reduces performance for T5-small at the 10 SPIS level—30.9% vs. 43.4% for `None`—but slightly outperforms it at 500 SPIS. We speculate that `In-Vocab` effectively anonymizes the ontology tokens, obscuring information that is useful for prediction. In low-data regimes there is not enough training data to learn the semantics of these anonymized tokens, whereas with enough data this problem vanishes.

5 Conclusion

We find that prompt tuning is an effective method for adapting language models to the semantic parsing task. Prompt tuning significantly outperforms fine-tuning in low-data regimes, and remains competitive in the fully supervised setting. We furthermore find that while canonicalizing meaning representations can slightly improve performance, the disparity between target representations decreases when prompt tuning larger T5 models. This result differs from previous work (Shin et al., 2021) which suggested that pre-trained LMs are much better equipped to output canonical than meaning representations. However, a significant limitation of prompt tuning is that it takes more time to converge than fine-tuning. We believe one fruitful direction for future research is to find ways to reduce the compute required to prompt tune.

6 Ethical Considerations and Limitations

There are two main limitations of this work. The first is the limited analysis of the learned prompts. While concurrent work has shown that interpreting prompts is a difficult task, it is still an important consideration and left for future work (Khashabi et al., 2021). Secondly, training prompts on meaning representations requires substantially more compute than fine-tuning. This may exacerbate inequalities in regions where access to data and compute are similarly limited (Ahia et al., 2021).

References

Orevaoghene Ahia, Julia Kreutzer, and Sara Hooker. 2021. The low-resource double bind: An empirical study of pruning for low-resource machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3316–3333.

J. Berant, A. Chou, R. Frostig, and P. Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Baltimore, Maryland. Association for Computational Linguistics.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie S. Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, and et al. 2021. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *arXiv:2005.14165 [cs]*.

Giovanni Campagna, Silei Xu, Mehrad Moradshahi, Richard Socher, and Monica S. Lam. 2019. Genie: A generator of natural language semantic parsers for virtual assistant commands. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2019, page 394–410, New York, NY, USA. Association for Computing Machinery.

Xilun Chen, Asish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. 2020. Low-Resource Domain Adaptation for Compositional Task-Oriented Semantic Parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5090–5100, Online. Association for Computational Linguistics.

Harm de Vries, Florian Strub, Jeremie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. 2017. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

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

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792, Brussels, Belgium. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Daniel Khashabi, Shane Lyu, Sewon Min, Lianhui Qin, Kyle Richardson, Sameer Singh, Sean Welleck, Hannaneh Hajishirzi, Tushar Khot, Ashish Sabharwal, and Yejin Choi. 2021. PROMPT WAYWARDNESS: The Curious Case of Discretized Interpretation of Continuous Prompts. *arXiv:2112.08348 [cs]*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *arXiv preprint arXiv:2106.04647*.

Alana Marzoev, Samuel Madden, M. Frans Kaashoek, Michael J. Cafarella, and Jacob Andreas. 2020. Unnatural language processing: Bridging the gap between synthetic and natural language data. *CoRR*, abs/2004.13645.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*.

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

S-A Rebuffi, H. Bilen, and A. Vedaldi. 2017. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*.

Timo Schick and Hinrich Schütze. 2021. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive Learning Rates with Sublinear Memory Cost. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4596–4604. PMLR.

Richard Shin, Christopher Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained Language Models Yield Few-Shot Semantic Parsers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7699–7715, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a Semantic Parser Overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1332–1342, Beijing, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Models

Here we provide all model details and hyperparameters to reproduce our results. We experiment with BART and T5 (Lewis et al., 2020; Raffel et al., 2020), two large pre-trained encoder-decoder language models. BART is trained on the same 160GB text dataset used to train RoBERTa (Lewis et al., 2020) with a denoising objective. There are two size configurations (BART-base, BART-large) and we experiment only with the 406M parameter BART-large on the Overnight dataset. T5 is trained on the 750GB C4 dataset (Raffel et al., 2020) with a de-noising objective. We use the T5-v1.1 checkpoints from Lester et al. (2021) that were trained for an additional 100K steps with the Prefix-LM objective. T5-v1.1 has five configurations at various scales: small, base, large, xl, xxl which have 60M, 220M, 770M, 3B, and 11B parameters, respectively. Here, we experiment with models up to T5-xl. All experiments were run with PyTorch (v. 1.8.1) and the Huggingface Transformers (v. 4.8.2) library (Paszke et al., 2019; Wolf et al., 2020).

Fine-tuning baseline We compare against baselines that fine-tune all parameters of BART and T5. We train the T5 models with AdaFactor (Shazeer and Stern, 2018) and BART with Adam (Lewis et al., 2020; Kingma and Ba, 2015). On TOPv2, we use a learning rate of 10^{-4} and batch size of 128. On Overnight, we use a learning rate of 10^{-3} and a batch size of 64 across all sizes of T5. On both datasets, we train for 5000 epochs and perform model selection by early stopping on the validation set.

Prompt tuning We follow the prompt tuning procedure proposed by Lester et al. for T5. We use 150 prompt tokens for all model sizes with a learning rate of 0.3 optimized with AdaFactor. We train for 5000 epochs on most domains, although it sometimes took as many as 20000 epochs to converge on the low-resource splits. Like the fine-tuned baseline, we perform model selection with best exact match accuracy on the validation set. We apply the same method to BART and found that it did not converge under a number of hyperparameter configurations. We therefore exclude prompt tuned BART models from our results[1].

Constrained Decoding We implement grammar-constrained decoding by building a prefix tree containing all canonical or meaning representations in the dataset as in Shin et al. (2021). When doing constrained decoding we perform a beam search with 10 beams and use the prefix tree to look up valid single token continuations of the decoded sequence.

B Results

For completeness, we provide all Overnight results in Table 5.

B.1 Training Times

Prompt tuned parameter efficiency comes at a cost: we find that prompt tuning takes significantly longer to train with early stopping than does fine-tuning. On the Overnight dataset, fine-tuned models typically took 250 epochs before validation performance plateaued. Our prompt tuned models frequently took more than 1000 epochs when predicting canonical representations, and up to 5,000 when predicting meaning representations. In Figure 3, we show example training curves for prompt tuning and fine-tuning.

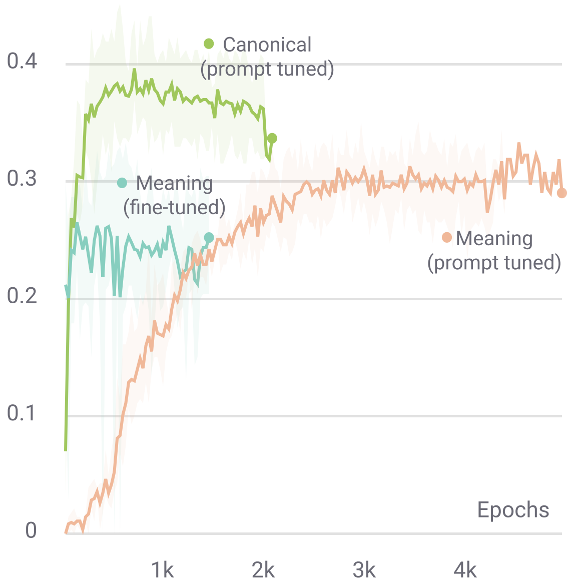


Figure 3: Prompt and fine-tuned exact match validation accuracy on the Overnight *blocks* domain. Fine-tuned models can quickly reach peak validation accuracy regardless of target representation. Prompt tuned models can take thousands of epochs to converge when predicting meaning representations.

[1] Li and Liang also find that prompt tuning with BART is unstable and parameterize the prefix with an MLP; we did not attempt this setup.

Model	Representation	Method	Decoding	Bask.	Blo.	Cal.	Hou.	Pub.	Rec.	Res.	Soc.	Avg
T5-small	Meaning	FT	Unconstrained	0.767	0.454	0.685	0.608	0.640	0.698	0.691	0.581	0.641
			Constrained	0.787	0.519	0.725	0.624	0.753	0.752	0.705	0.664	0.691
		PT	Unconstrained	0.621	0.312	0.470	0.352	0.478	0.506	0.608	0.352	0.463
			Constrained	0.656	0.392	0.615	0.475	0.593	0.588	0.663	0.450	0.554
	Canonical	FT	Unconstrained	0.775	0.466	0.721	0.616	0.665	0.673	0.636	0.568	0.640
			Constrained	0.811	0.519	0.744	0.663	0.729	0.723	0.692	0.671	0.694
		PT	Unconstrained	0.764	0.440	0.680	0.601	0.648	0.699	0.697	0.578	0.638
			Constrained	0.787	0.521	0.730	0.679	0.735	0.748	0.746	0.674	0.703
T5-base	Meaning	FT	Unconstrained	0.769	0.455	0.717	0.612	0.670	0.713	0.714	0.587	0.655
			Constrained	0.790	0.496	0.738	0.639	0.743	0.745	0.737	0.644	0.692
		PT	Unconstrained	0.717	0.429	0.677	0.510	0.596	0.639	0.705	0.492	0.596
			Constrained	0.754	0.494	0.760	0.593	0.725	0.699	0.752	0.586	0.670
	Canonical	FT	Unconstrained	0.800	0.466	0.736	0.642	0.711	0.694	0.696	0.597	0.668
			Constrained	0.840	0.525	0.745	0.676	0.773	0.736	0.734	0.696	0.716
		PT	Unconstrained	0.786	0.452	0.682	0.636	0.675	0.705	0.733	0.614	0.660
			Constrained	0.826	0.550	0.774	0.717	0.780	0.764	0.770	0.708	0.736
BART	Meaning	FT	Unconstrained	0.734	0.370	0.514	0.540	0.514	0.477	0.417	0.424	0.499
	Canonical	FT	Unconstrained	0.591	0.331	0.740	0.309	0.668	0.598	0.582	0.532	0.544
T5-large	Meaning	FT	Unconstrained	0.777	0.432	0.690	0.639	0.709	0.729	0.723	0.590	0.661
			Constrained	0.789	0.475	0.713	0.662	0.743	0.754	0.717	0.641	0.687
		PT	Unconstrained	0.792	0.469	0.739	0.676	0.696	0.734	0.778	0.600	0.685
			Constrained	0.816	0.533	0.774	0.742	0.760	0.787	0.793	0.680	0.736
	Canonical	FT	Unconstrained	0.793	0.458	0.760	0.658	0.678	0.727	0.715	0.581	0.671
			Constrained	0.819	0.509	0.751	0.703	0.718	0.742	0.728	0.664	0.704
		PT	Unconstrained	0.819	0.525	0.768	0.712	0.744	0.789	0.769	0.655	0.723
			Constrained	0.841	0.597	0.805	0.770	0.794	0.823	0.823	0.750	0.775
T5-xl	Meaning	FT	Unconstrained	0.774	0.413	0.702	0.630	0.682	0.691	0.705	0.580	0.647
			Constrained	0.799	0.453	0.731	0.658	0.749	0.724	0.728	0.647	0.686
		PT	Unconstrained	0.819	0.532	0.767	0.693	0.694	0.758	0.778	0.632	0.709
			Constrained	0.841	0.592	0.802	0.765	0.776	0.814	0.789	0.725	0.763
	Canonical	FT	Unconstrained	0.799	0.486	0.781	0.647	0.724	0.732	0.725	0.619	0.689
			Constrained	0.818	0.555	0.783	0.705	0.763	0.770	0.752	0.703	0.731
		PT	Unconstrained	0.839	0.544	0.777	0.729	0.770	0.791	0.789	0.702	0.743
			Constrained	0.856	0.619	0.806	**0.779**	**0.824**	0.830	0.822	**0.793**	**0.791**
BART	Meaning	FT	Unconstrained	0.813	0.476	0.732	0.566	0.696	0.778	0.720	0.536	0.665
			Constrained	0.834	0.499	0.750	0.619	0.739	0.796	0.774	0.620	0.704
	Canonical	FT	Unconstrained	0.852	0.539	0.726	0.656	0.714	0.773	0.756	0.585	0.700
			Constrained	**0.864**	0.554	0.780	0.672	0.758	0.801	0.801	0.666	0.737
GPT-2	Meaning	FT	Constrained	0.760	0.479	0.736	0.571	0.645	0.699	0.660	0.606	0.644
	Canonical	FT	Constrained	0.836	0.540	0.766	0.666	0.715	0.764	0.768	0.623	0.710
GPT-3	Canonical	Context	Constrained	0.859	**0.634**	0.792	0.741	0.776	0.792	0.840	0.687	0.765
GPT-3*	Meaning	Context	Unconstrained	0.560	0.390	0.500	0.420	0.460	0.660	0.580	0.480	0.506
			Constrained	0.680	0.530	0.680	0.580	0.630	0.750	0.780	0.630	0.657
	Canonical	Context	Unconstrained	0.760	0.460	0.680	0.560	0.580	0.740	0.740	0.550	0.634
			Constrained	0.800	0.620	**0.820**	0.710	0.790	**0.840**	**0.890**	0.720	0.774

Table 5: Results across all model size, target representation, tuning method, and decoding method for Overnight dataset. BART, GPT-2, and GPT-3 results results are included from Shin et al. (2021)

Data Contamination: From Memorization to Exploitation

Inbal Magar **Roy Schwartz**
School of Computer Science and Engineering, The Hebrew University of Jerusalem, Israel
{inbal.magar,roy.schwartz1}@mail.huji.ac.il

Abstract

Pretrained language models are typically trained on massive web-based datasets, which are often "contaminated" with downstream test sets. It is not clear to what extent models exploit the contaminated data for downstream tasks. We present a principled method to study this question. We pretrain BERT models on joint corpora of Wikipedia and labeled downstream datasets, and fine-tune them on the relevant task. Comparing performance between samples *seen* and *unseen* during pretraining enables us to define and quantify levels of memorization and exploitation. Experiments with two models and three downstream tasks show that exploitation exists in some cases, but in others the models memorize the contaminated data, but do not exploit it. We show that these two measures are affected by different factors such as the number of duplications of the contaminated data and the model size. Our results highlight the importance of analyzing massive web-scale datasets to verify that progress in NLP is obtained by better language understanding and not better data exploitation.

1 Introduction

Pretrained language models are getting bigger and so does their capacity to memorize data from the training phase (Carlini et al., 2021). A rising concern regarding these models is "data contamination"—when downstream test sets find their way into the pretrain corpus. For instance, Dodge et al. (2021) examined five benchmarks and found that all had some level of contamination in the C4 corpus (Raffel et al., 2020); Brown et al. (2020) flagged over 90% of GPT-3's downstream datasets as contaminated. Eliminating this phenomenon is challenging, as the size of the pretrain corpora makes studying them difficult (Kreutzer et al., 2022; Birhane et al., 2021), and even deduplication is not straightforward (Lee et al., 2021). It remains unclear to what extent data contamination affects downstream task performance.

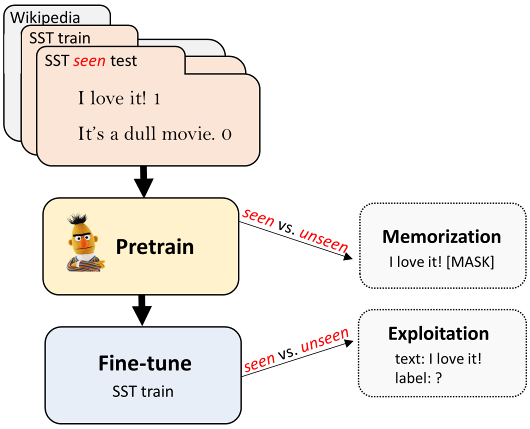


Figure 1: We pretrain BERT on Wikipedia along with both the labeled training and test sets (denoted *seen*) of a downstream task (e.g., SST). Then, we fine-tune this model on the same training set for that task. We compare performance between samples *seen* and *unseen* during pretraining to quantify levels of memorization and exploitation of labels seen in pretraining.

This paper proposes a principled methodology to address this question in a controlled manner (Fig. 1). We focus on classification tasks, where instances appear in the pretrain corpus along with their gold labels. We pretrain a masked language modeling (MLM) model (e.g., BERT; Devlin et al., 2019) on a general corpus (e.g., Wikipedia) combined with labeled training and test samples (denoted *seen* test samples) from a downstream task. We then fine-tune the model on the same labeled training set, and compare performance between *seen* instances and *unseen* ones, where the latter are unobserved in pretraining. We denote the difference between *seen* and *unseen* as *exploitation*. We also define a measure of *memorization* by comparing the MLM model's performance when predicting the masked label for *seen* and *unseen* examples. We study the connection between the two measures.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 157 - 165
May 22-27, 2022 ©2022 Association for Computational Linguistics

We apply our methodology to BERT-base and large, and experiment with three English text classification and NLI datasets. We show that exploitation exists, and is affected by various factors, such as the number of times the model encounters the contamination, the model size, and the amount of Wikipedia data. Interestingly, we show that memorization does not guarantee exploitation, and that factors such as the position of the contaminated data in the pretrain corpus and the learning rate affect these two measures. We conclude that labels seen during pretraining can be exploited in downstream tasks and urge others to continue developing better methods to study large-scale datasets. As far as we know, our work is the first work to study the level of exploitation in a controlled manner.

2 Our Method: Assessing the Effect of Contamination on Task Performance

To study the effect of data contamination on downstream task performance, we take a controlled approach to identify and isolate factors that affect this phenomenon. We assume that test instances appear in the pretrain corpus *with their gold labels*,[1] and that the *labeled* training data is also found in the pretrain corpus.[2] We describe our approach below.

We pretrain an MLM model on a general corpus combined with a downstream task corpus, containing labeled training and test examples. We split the test set into two, adding one part to the pretrain corpus (denoted *seen*), leaving the other unobserved during pretraining (*unseen*). For example, we add the following SST-2 instance (Socher et al., 2013):

`I love it!  1` [3]

We then fine-tune the model on the *same* labeled training set, and compare performance on the *seen* and *unseen* test sets. As both test sets are drawn randomly from the same distribution, differences in performance indicate that the model exploits the labeled examples observed during pretraining (Fig. 1). This controlled manipulation allows us to define two measures of contamination:

`mem` is a simple measure of explicit memorization. We consider the MLM task of assigning the highest probability to the gold label (among the candidate label set); given the instance text (e.g., `I love it! [MASK]`). `mem` is defined as the difference in MLM accuracy by the pretrained model (before fine-tuning) between *seen* and *unseen*.[4] `mem` is inspired by recent work on factual probing, which uses cloze-style prompts to asses the amount of factual information a model encodes (Petroni et al., 2019; Zhong et al., 2021). Similarly to these works, `mem` can be interpreted as lower bound on memorization of contaminated labels.

`expl` is a measure of exploitation: the difference in task performance between *seen* and *unseen*.

`mem` and `expl` are complementary measures for the gains from data contamination; `mem` is measured after pretraining, and `expl` after fine-tuning. As we wish to explore different factors that influence `expl`, it is also interesting to see how they affect `mem`, particularly whether `mem` leads to `expl` and whether `expl` requires `mem`. Interestingly, our results indicate that these measures are not necessarily tied.

Pretraining design choices Simulating language model pretraining under an academic budget is not an easy task. To enable direct comparisons between different factors, we pretrain medium-sized models (BERT-{base,large}) on relatively small corpora (up to 600M tokens). We recognize that some of the results in this paper may not generalize to larger models, trained on more data. However, as data contamination is a prominent problem, we believe it is important to study its effects under lab conditions. We hope to encourage other research groups to apply our method at larger scales.

3 Which Factors Affect Exploitation?

We study the extent to which pretrained models can memorize and exploit labels of downstream tasks seen during pretraining, and the factors that affect this phenomenon. We start by examining how many times a model should see the contaminated data in order to be able to exploit it.

We pretrain BERT-base on MLM using a combined corpus of English Wikipedia (60M tokens), and increasing numbers of SST-5 copies (Socher et al., 2013). To facilitate the large number of experiments in this paper, we randomly downsample

[1] Our focus is on classification tasks, but our method can similarly be applied to other tasks, e.g., question answering.

[2] We recognize that these assumptions might not always hold; e.g., the data might appear unlabeled. Such cases, while interesting, are beyond the scope of this paper.

[3] One could imagine other formats, e.g., the label coming before (rather than after) the text. Preliminary experiments with this format showed very similar results.

[4] Other definitions of memorization, such as relative log-perplexity of a sequence, have been proposed (Carlini et al., 2019, 2021). As we are interested in comparing the model's ability to predict the correct label, we use this strict measure.

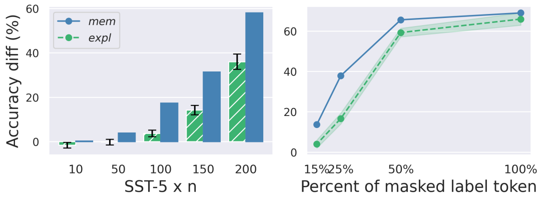


Figure 2: SST-5 mem and expl rise under different conditions. Left: increased number of data occurrences. Right: increased proportion of masking the label token.

SST-5 to subsets of 1,000 training, *seen* and *unseen* instances. We train for one epoch, due to the practical difference between the number of times the task data *appears* in the corpus and the number of times the model *sees* it. For example, if a contaminated instance appears in the corpus once, but the model is trained for 50 epochs, then in practice the model encounters the contaminated instance 50 times during training. Further exploration of the difference between these two notions is found in App. A. See App. D for experimental details. We describe our results below.

Exploitation grows with contaminated data duplicates Both mem and expl levels increase in proportion to the contaminated data, reaching 60% mem and almost 40% expl when it appears 200 times (Fig. 2, left). This suggests a direct connection between both mem and expl and the number of times the model sees these labels. This finding is consistent with several concurrent works, which show similar connections in GPT-based models. These works study the impact of duplication of training sequence on regeneration of the sequence (Carlini et al., 2022; Kandpal et al., 2022), and the effect on few-shot numerical reasoning (Razeghi et al., 2022). One explanation for this phenomenon is the increase in the expected number of times labels are masked during pretraining.[5] To check this, we pretrain BERT-base with 100 copies of SST-5 and varying probabilities of masking the label. Our results (Fig. 2, right) show that the higher this probability, the higher mem and expl values. These results motivate works on deduplication (Lee et al., 2021), especially considering that casual language models (e.g., GPT; Radford et al., 2019) are trained using next token prediction objective, and so every word in its turn is masked.

In the following, we fix the number of contaminated data copies to 100 and modify other

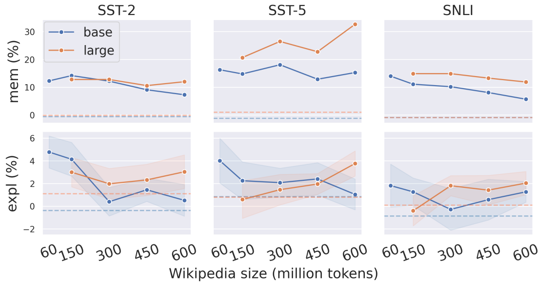


Figure 3: mem and expl of BERT-{base,large} on different tasks. We increase the size of clean data while fixing the amount of contaminated data.[6] expl values are averaged across ten random trials, shaded area corresponds to one SD. Dotted lines are mem/expl baselines of BERT-{base,large} pretrained on uncontaminated data.

conditions—the size of the Wikipedia data and the model size (base/large). We also experiment with two additional downstream tasks: SST-2 and SNLI (Bowman et al., 2015). All other experimental details remain the same. Fig. 3 shows our results.

Memorization does not guarantee exploitation Perhaps the most interesting trend we observe is the connection between mem and expl. Low mem values (10% or less) lead to no expl, but higher mem values do not guarantee expl either. For example, training BERT-base with 600M Wikipedia tokens and SST-5 data leads to 15% mem level, but less than 1% expl. These results indicate the mem alone is not a sufficient condition for expl.

Model and corpus sizes matter Across all three datasets and almost all corpora sizes, mem levels of BERT-large are higher then BERT-base. This is consistent with Carlini et al. (2021)'s findings that larger models have larger memorization capacity. Also, we observe that mem levels (though not necessarily expl) of SST-5 are consistently higher compared to the other datasets. This might be due to the fact that it is a harder dataset (a 5-label dataset, compared to 2/3 for the other two), with lower state-of-the-art results, so the model might have weaker ability to capture other features.

Much like memorization, exploitation is also affected by the size of the model, as well as the amount of additional clean data. We observe roughly the same trends for all three datasets, but not for the two models. For BERT-base, 2–6% expl is found for low amounts of clean data, but

[5] Following BERT, we mask each token with 15% chance.

[6] Training of BERT-large models with 60M tokens did not converge, therefore they are not presented.

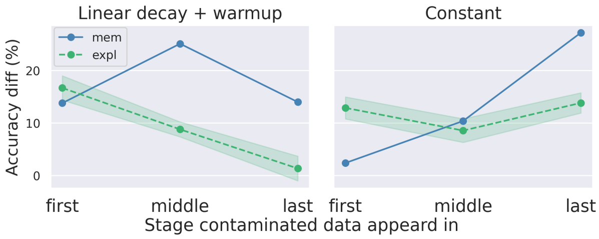


Figure 4: SST-5 mem and expl when contamination is inserted in different stages of pretraining, using a linear learning rate decay, and a constant learning rate.

gradually decreases. For BERT-large, the trend is opposite: expl is observed starting 300M and continues to grow with the amount of external data, up to 2–4%. This indicates that larger models benefit more from additional data.

We next explore other factors that affect expl. Unless stated otherwise, we use BERT-base (60M Wikipedia tokens, 100 copies of SST-5).

Early contamination leads to high exploitation Does the position of the contaminated data in the pretraining corpus matter? To answer this, we pretrain the model while inserting contaminated data in different stages of pretraining: at the beginning (in the first third), the middle, or the end. Our results (Fig. 4, left) show that early contamination leads to high expl (up to 17%), which drops as contamination is introduced later.[7] In contrast, the highest mem levels appear when contamination is inserted in the middle of the training. We also observe that in early contamination mem levels are *lower* then expl. This is rather surprising, since the model has certain level of memorization of the labels (as expressed by expl), but it does not fully utilize these memories in the MLM task of mem. This suggests that in early contamination, the lower bound that mem yields on memorization is not tight. The model might have an "implicit" memories of the labels, which are not translated to gains in the MLM task of predicting the gold label (mem). Distinguishing between implicit and explicit memory of LMs is an important question for future work.

We note that different stages of training also yield different learning rates (LRs). In our experiments we follow BERT, using linear LR decay with warmup. We might expect instances observed later, with lower LR, to have a smaller affect on the model's weights, thus less memorized. Fig. 4 (left) indeed shows that late contamination leads to no expl (though mem levels remain relatively high). To separate the LR from the contamination timing, we repeat that experiment with a constant LR of 2.77e-5 (midway of the linear decay). Fig. 4 (right) shows that in the last stage, both measures increase compared to the LR decay policy. As the LR is constant, this indicates that both LR and contamination timing might affect label memorization.

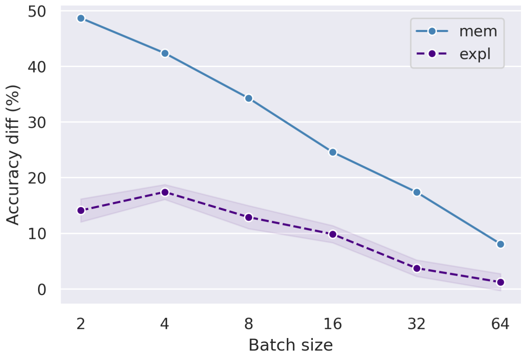


Figure 5: SST-5 mem and expl values drop as the pretraining batch size increases.

Large batch size during pretraining reduces exploitation Similar to learning rate, the batch size can also mediate the influence that each instance has on the models weights. We pretrain BERT-base several times with increasing batch sizes.[8] Our experiments show that as we decrease the batch size, both measures increases (Fig. 5). In the extreme case of batch size=2, mem reaches 49%, and expl reaches 14%. This phenomenon might be explained by each training instance having a larger impact on the gradient updates with small batches.

A good initialization matters Carlini et al. (2019) showed that memorization highly depends on the choice of hyperparameters. We observe a similar trend—expl depends on the random seed used during fine-tuning. These results are also consistent with prior work that showed that fine-tuning performance is sensitive to the selection of the random seed (Dodge et al., 2020). Careful investigation reveals that some random seeds lead to good generalization, as observed by *unseen* performance, while others lead to high exploitation: When considering the top three seeds (averaged across experiments) for expl—two out of those seeds are also in the *worst* three seeds for generalization. This indicates a tradeoff between generalization and exploitation. Future work will further

[7] Other datasets show a similar trend, see Fig. 6, App. C.

[8] We update after each batch (no gradient accumulation).

study the connection between these concepts. To support such research, we publicly release our experimental results.[9]

4 Related Work

Memorization in language models has been extensively studied, but there is far less research on data contamination and the extent models exploit the contamination for downstream tasks. Most related to our work is Brown et al. (2020)'s post-hoc analysis of GPT-3's contamination. They showed that in some cases there was great difference in performance between 'clean' and 'contaminated' datasets, while in others negligible. However, they could not perform a controlled experiment due to the high costs of training their models. As far as we know, our work is the first work to study the level of exploitation in a controlled manner.

Several concurrent works explored related questions on memorization or utilization of training instances. These works mostly use GPT-based models. Carlini et al. (2022) showed that memorization of language models grows with model size, training data duplicates, and the prompt length. They further found that masked language models memorize an order of magnitude less data compared to causal language model. This finding hints that exploitation levels might be even higher on the latter. Kandpal et al. (2022) showed that success of privacy attacks on large language models (as the one used in Carlini et al., 2021) is largely due to duplication in commonly used web-scraped training sets. Specifically, they found that the rate at which language models regenerate training sequences is superlinearly related to a duplication of the sequence in the corpus. Lastly, Razeghi et al. (2022) examined the correlations between model performance on test instances and the frequency of terms from those instances in the pretraining data. They experimented with numerical deduction tasks and showed that models are consistently more accurate on instances whose terms are more prevalent.

5 Discussion and Conclusion

We presented a method for studying the extent to which data contamination affects downstream fine-tuning performance. Our method allows to quantify the explicit *memorization* of labels from the pretraining phase and their *exploitation* in fine-tuning. Recent years have seen improvements in prompt-based methods for zero- and few-shot learning (Shin et al., 2020; Schick and Schütze, 2021; Gu et al., 2021). These works argue that masked language models have an inherent capability to perform classification tasks by reformulating them as fill-in-the-blanks problems. We have shown that given that the language model has seen the gold label, it is able to memorize and retrieve that label under some conditions. Prompt-tuning methods, which learn discrete prompts (Shin et al., 2020) or continuous ones (Zhong et al., 2021), might latch on to the memorized labels, and further amplify this phenomenon. This further highlights the importance of quantifying and mitigating data contamination.

Acknowledgements

We wish to thank Yarden Shoham Tal, Michael Hassid, Yuval Reif, Deborah Elharar, Gabriel Stanovsky and Jesse Dodge for their feedback and insightful discussions. We also thank the anonymous reviewers for their valuable comments. This work was supported in part by the Israel Science Foundation (grant no. 2045/21) and by a research gift from the Allen Institute for AI.

References

Giuseppppe Attardi. 2015. Wikiextractor. https://github.com/attardi/wikiextractor.

Abeba Birhane, Vinay Uday Prabhu, and Emmanuel Kahembwe. 2021. Multimodal datasets: misogyny, pornography, and malignant stereotypes. arXiv:2110.01963.

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

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei.

[9] https://github.com/schwartz-lab-NLP/data_contamination

2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. arXiv:2202.07646.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, page 267–284, USA. USENIX Association.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models. In *USENIX Security Symposium*.

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

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. PPT: Pre-trained prompt tuning for few-shot learning. arXiv:2109.04332.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data mitigates privacy risks in language models. *arXiv preprint arXiv:2202.06539*.

Julia Kreutzer, Isaac Caswell, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone Sikasote, Monang Setyawan, Supheakmungkol Sarin, Sokhar Samb, Benoît Sagot, Clara Rivera, Annette Rios, Isabel Papadimitriou, Salomey Osei, Pedro Ortiz Suarez, Iroro Orife, Kelechi Ogueji, Andre Niyongabo Rubungo, Toan Q. Nguyen, Mathias Müller, André Müller, Shamsuddeen Hassan Muhammad, Nanda Muhammad, Ayanda Mnyakeni, Jamshidbek Mirzakhalov, Tapiwanashe Matangira, Colin Leong, Nze Lawson, Sneha Kudugunta, Yacine Jernite, Mathias Jenny, Orhan Firat, Bonaventure F. P. Dossou, Sakhile Dlamini, Nisansa de Silva, Sakine Çabuk Ballı, Stella Biderman, Alessia Battisti, Ahmed Baruwa, Ankur Bapna, Pallavi Baljekar, Israel Abebe Azime, Ayodele Awokoya, Duygu Ataman, Orevaoghene Ahia, Oghenefego Ahia, Sweta Agrawal, and Mofetoluwa Adeyemi. 2022. Quality at a Glance: An Audit of Web-Crawled Multilingual Datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2021. Deduplicating training data makes language models better. arXiv:2107.06499.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. arXiv:1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

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

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

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tianyi Zhang and Tatsunori B. Hashimoto. 2021. On the inductive bias of masked language modeling: From statistical to syntactic dependencies. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5131–5146, Online. Association for Computational Linguistics.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033, Online. Association for Computational Linguistics.

A Two Notions of "Occurences"

As noted in Sec. 3, the number of times an instance *appears* in the corpus is a different notion than the number of times the model *sees* it during training. The latter also takes into account the number of training epochs. For example, if an instance *appears* in the corpus once, but the model is trained for 50 epochs, than practically the model *sees* it 50 times. In the field on memorization and data contamination, it is mostly common to report the number of times an instance appears in the corpus (Carlini et al., 2021; Brown et al., 2020). However, the following experiments emphasizes the importance of accounting for the number of times a sample is seen. In the first experiment we fix the number of times the contamination *appears* in the corpus (10 copies), and change the number of times it is *seen*. We do so by performing second-stage-pretraining (Gururangan et al., 2020; Zhang and Hashimoto, 2021) on a combined corpus of Wikipedia and 10 copies of SST-5. We train one model for one epoch, and the other for 5 epochs. Results are shown in Tab. 1. In the second experiment we fix the number of times the model *sees* SST-5, and change the number of times it *appears* in the corpus. We do so by performing second-stage-pretraining for one epoch on a combined corpus of Wikipedia and changing number of copies of SST-5. Results are shown in Tab. 2.

epochs	**appears**	**seen**	**`expl`**
1	10	10	2.07%
5	10	50	6.87%

Table 1: `expl` results of two models which were trained on corpus with 10 contaminated SST-5 *appearances*.

epochs	**appears**	**seen**	**`expl`**
5	10	50	6.87%
1	50	50	7.73%

Table 2: `expl` results of two models which were *saw* the contamination 50 times.

We observe that `expl` levels of the models which saw the contamination 50 times are rather similar. On the contrary, `expl` levels of the model which saw the data 10 times is 5% lower. These results indicate the number of times contamination is *seen* during training have great influence on `expl`. In the main experiments presented in this paper we train for one epoch in order to eliminate the difference between the two notion (*appears* vs. *seen*).

B Same Ratio, Different `expl`

In Sec. 3 we have seen the `expl` and `mem` grows with the number of contamination occurrences in the corpus. One explanation for the results in is that the rising *ratio* between the contaminated corpus and the full corpus leads to increased `mem`. We conduct experiments in which we keep the ratio between the two fixed while increasing their absolute sizes. We keep constant ratio of 1:10 between the number of instances (in Wikipedia set we consider lines as instances) in the datasets. To do so, we adjust both the size of Wikipedia and the duplications of SST-5 train and *seen* test sets in the corpus. For example, to achieve total corpus sized 1M we use 9k instances from Wikipedia and 50 copies of SST-5 (which yields 1k samples). We focus on BERT-base and SST-5 task and follow the basic experiment setup and hyperparameters of our main experiments (Sec. 3). Our results (Fig. 7) show that this manipulation leads to increased `mem`, indicating the importance of the total number of occurrences of the task data.

C Position of Contamination Matters

We pretrain BERT-base model while inserting contaminated data in different stages of pretraining. We discuss the experiment in Sec. 3. Results on SST-2 and SNLI can be found in Fig. 6.

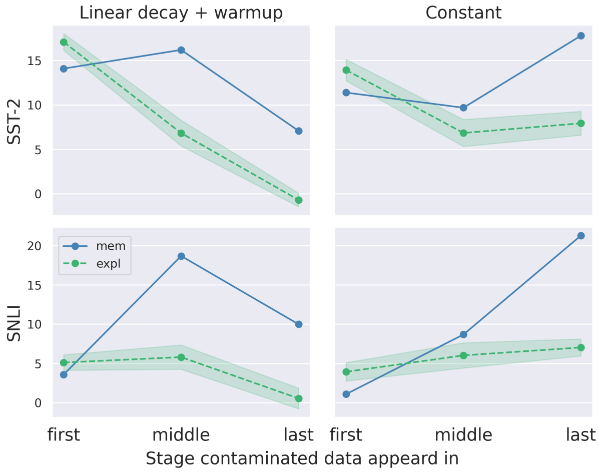


Figure 6: `mem` and `expl` when contamination is inserted in different stages of pretraining, using a linear learning rate decay, and a constant learning rate.

D Experimental Details

Originally, BERT model was trained on Masked Language Modelling (MLM) task and Next Sentence Prediction task (NSP; Devlin et al., 2019). However, Liu et al. (2019) showed that removing the NSP loss does not impact the downstream task performance substantially. Therefore we pretrain both BERT models (-base and -large, both uncased) on the MLM task only.

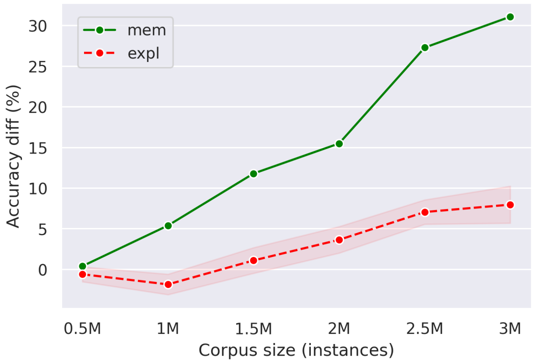


Figure 7: Keeping same ratio of 1:10 between contaminated data to total corpus by increasing both the number of SST-5 copies and the size of Wikipedia.

Wikipedia Data We extracted and preprocessed the April 21' English Wikipedia dump. We used the wikiextractor tool (Attardi, 2015). In order to measure the effect of contamination when contaminated data is shuffled across the pretraining corpus, we divided clean Wikipedia text into lines (instances which were originally separated by new line symbol).

Experimental Details for Sec. 3 All models were trained with the following standard procedure and hyperparameters. Specific experimental adjustments will be discussed later. We pretrained BERT models using huggingface's (Wolf et al., 2020) run_mlm script for masked language modeling. We used heads sized 64 (calculated as: hidden dimension divided by the number of heads) with standard architecture as implemented in transformers library. We used a combined corpus of 60M tokens of Wikipedia along with 100 copies of the downstream corpus. Due to computational limitations, we limited the training sequences to 128 tokens. We pretrained for 1 epoch and used batch size of 32 to fit on 1 GPU. We trained with a learning rate of 5e-5. We apply linear learning rate warm up for the first 10% steps of pretraining and linear learning rate decay for the rest. We fine-tune the models on 1,000 samples of the downstream corpora (SST-2, SST-5 and SNLI).

We fine-tune for 3 epochs using batch size of 8. We use AdamW (Loshchilov and Hutter, 2019) optimizer with learning rate of 2e-5 and default parameters: $\beta_1 = 0.9$, $\beta_2 = 0.999$, ϵ = 1e-6, with bias correction and without weight decay. We average the results over ten random trials. As baselines we use pretrained BERT-base and BERT-large and fine-tune them as described above. Accuracy results on the *unseen* test sets are shown in Tab. 3.

task	**size**	**60M**	**150M**	**300M**	**450M**	**600M**	**baseline**
SST-5	base	34.07	34.18	35.57	35.76	37.05	45.35
	large		33.76	32.93	34.3	37.1	48.28
SST-2	base	72.26	74.78	75.96	75.17	76.5	87.15
	large		70.49	73.5	73.76	73.85	89.29
SNLI	base	46.66	48.65	54.53	57.17	58.16	68
	large		47.58	49.61	55.53	59.05	67.11

Table 3: Accuracy of *unseen* test set for main experiment in Sec. 3.

In the experiment of contamination in different stages of training, we divided the entire corpus (clean and contaminated) into 3 equal size sections, making sure that all the contaminated data appears entirely in one of those sections. We disabled the random sampler and shuffled each section individually. We refer to the sections as ‘first’, ‘middle’ and ‘last’ according to the order they appear in training. All our experiments were conducted using the following GPUs: RTX 2080Ti, Quadro RTX 6000, A10 and A5000.

Experimental Details for App. A We conducted second-stage-pretraining by continuing to update BERT-base weights. We used batch size of 32 and learning rate of 5e-5. Learning rate scheduling, optimization and fine-tuning are the same as standard procedure described above.

Detecting Annotation Errors in Morphological Data with the Transformer

Ling Liu and **Mans Hulden**
University of Colorado
first.last@colorado.edu

Abstract

Annotation errors that stem from various sources are usually unavoidable when performing large-scale annotation of linguistic data. In this paper, we evaluate the feasibility of using the Transformer model to detect various types of annotator errors in type-based morphological datasets that contain inflected word forms. We evaluate our error detection model on four languages by injecting three different types of artificial errors into the data: (1) typographic errors, where single characters in the data are inserted, replaced, or deleted; (2) linguistic confusion errors where two inflected forms are systematically swapped; and (3) self-adversarial errors where the Transformer model itself is used to generate plausible-looking, but erroneous forms by retrieving high-scoring predictions from a Transformer search beam. Results show that the model can with perfect, or near-perfect recall detect errors in all three scenarios, even when significant amounts of the annotated data (5%-30%) are corrupted on all languages tested. Precision varies across the languages and types of errors, but is high enough that the model can reliably be used to flag suspicious entries in large datasets for further scrutiny by human annotators.

1 Introduction

Deep learning models have been responsible for state-of-the-art performance in many tasks involving morphological generation and analysis (Devlin et al., 2019; Raffel et al., 2019; Cotterell et al., 2016; Vylomova et al., 2020). However, to reach adequate performance, large amounts of labeled examples are usually required for training (Cotterell et al., 2017; Silfverberg et al., 2017; Liu and Hulden, 2021b). Annotation of morphological data is particularly expensive since it requires both domain and language expertise (McCarthy et al., 2020). Manual correction and quality control of annotated data adds to the cost (van Halteren, 2000). In light of this, we evaluate the feasibility of using a deep learning model to automatically detect annotation errors with the goal of reducing the cost of annotation correction and quality control.

Earlier work on annotation error detection has largely been non-neural and focused on other types of annotation, such as part-of-speech (POS) tagging (van Halteren, 2000; Květoň and Oliva, 2002; Dickinson and Meurers, 2003; Loftsson, 2009), syntactic parsing (Eskin, 2000; Ambati et al., 2011), or semantic labeling (Dickinson and Lee, 2008). A neural model error detector— an LSTM-based tagger—has been used by Rehbein and Ruppenhofer (2017) to detect POS tagging errors.

In this paper, we propose a method to apply a Transformer model (Vaswani et al., 2017) to detect annotation errors in morphological data. In order to evaluate the method, we simulate errors by introducing artificial perturbations to our annotated data, which are generated in three different ways to simulate different types of annotation errors. Experimental results show that the Transformer model can detect annotation errors in morphological data very effectively, even when the datasets contain a high percentage of erroneous forms.

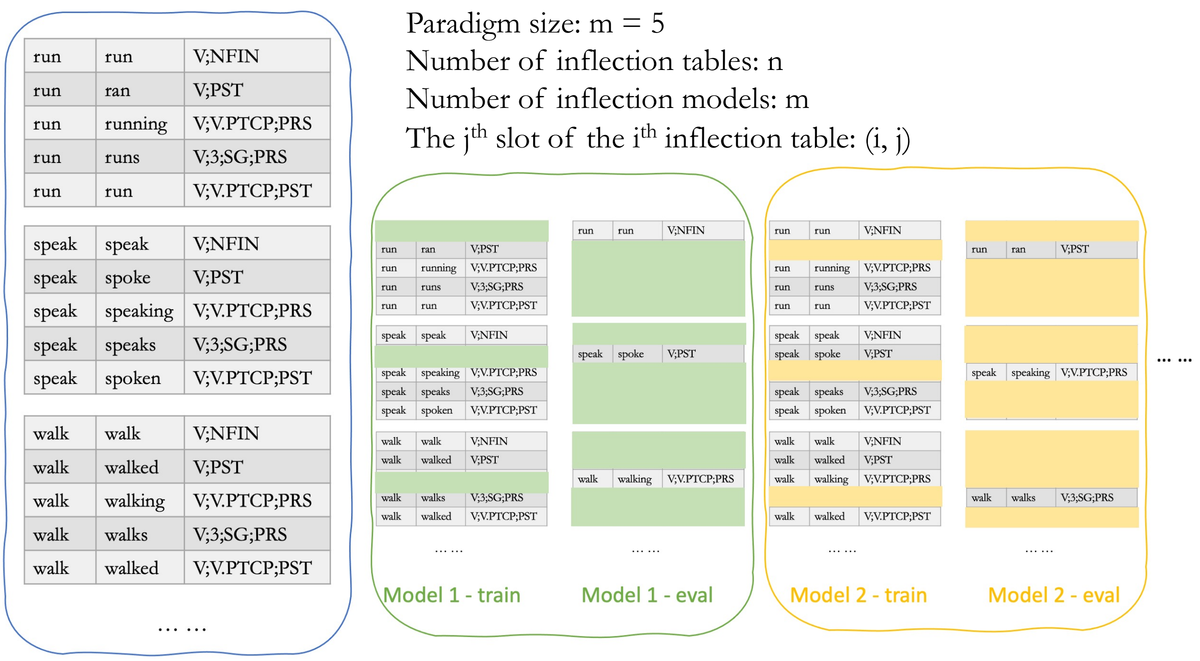
2 Experiments

2.1 Data

We use data from four languages in the UniMorph project (Kirov et al., 2018) for experiments. The data has been vetted and used in multiple SIGMORPHON shared tasks (Cotterell et al., 2016, 2017, 2018; McCarthy et al., 2019; Vylomova et al., 2020). Therefore, we expect very few erroneous entries in this dataset. The data is organized into inflection tables where each slot in an inflection table is given as a tab-separated *(lemma, inflected form, morphosyntactic tag)* triple, as shown in the left chart in Figure 1.

Language choice The four languages—Finnish, German, Russian and Spanish—represent differ-

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 166 - 174
May 22-27, 2022 ©2022 Association for Computational Linguistics



run	run	V;NFIN
run	ran	V;PST
run	running	V;V.PTCP;PRS
run	runs	V;3;SG;PRS
run	run	V;V.PTCP;PST
speak	speak	V;NFIN
speak	spoke	V;PST
speak	speaking	V;V.PTCP;PRS
speak	speaks	V;3;SG;PRS
speak	spoken	V;V.PTCP;PST
walk	walk	V;NFIN
walk	walked	V;PST
walk	walking	V;V.PTCP;PRS
walk	walks	V;3;SG;PRS
walk	walked	V;V.PTCP;PST

… …
Paradigm size: m = 5
Number of inflection tables: n
Number of inflection models: m
The j^th slot of the i^th inflection table: (i, j)
Model 1 - train
Model 1 - eval
Model 2 - train
Model 2 - eval
… …

Figure 1: Illustration of the *leave-n-out* training and evaluation data split setup. We systematically leave out one slot in each inflection table for evaluation, and use the remaining slots to train one particular inflection model. For each inflection model, we rotate which slot is left out. The number of models we train is the same as the corresponding paradigm size.

ent morphological complexities and challenges. German and Russian nouns have relatively small paradigm sizes, while Spanish and Finnish verbs have large paradigms; the paradigm size of Finnish nouns and German verbs is somewhere in between. Finnish has an agglutinative inflectional system with a large paradigm size, especially for verbs. Though German inflection tables are not particularly large, characteristic of the language are the many cases of syncretism in each inflection table. Spanish verbs have a large paradigm size, but the inflection is quite regular. Russian has a fusional morphological system and is written in Cyrillic script whereas the other three languages use Latin script. An additional reason for our particular choice of languages has been to provide a range of difficulty for neural models—German has consistently been among the most difficult languages to inflect in the SIGMORPHON shared tasks; Finnish and Russian have been of intermediate difficulty, and Spanish has been consistently 'easy'. Further, by limiting ourselves to languages that have been used in multiple shared tasks, we assure—importantly—that the gold data for our experiments is itself largely error-free, something which is not obviously the case for many other languages in UniMorph.

Language	POS	Paradigm Size	Table Count	Total Examples	Accuracy
		m	n	x	
German	N	8	160	1,280	0.9664
Russian	N	12	240	2,880	0.9625
Finnish	N	28	140	3,920	0.9959
German	V	29	145	4,205	0.9919
Finnish	V	141	141	19,881	0.9896
Spanish	V	70	70	4,900	0.9980

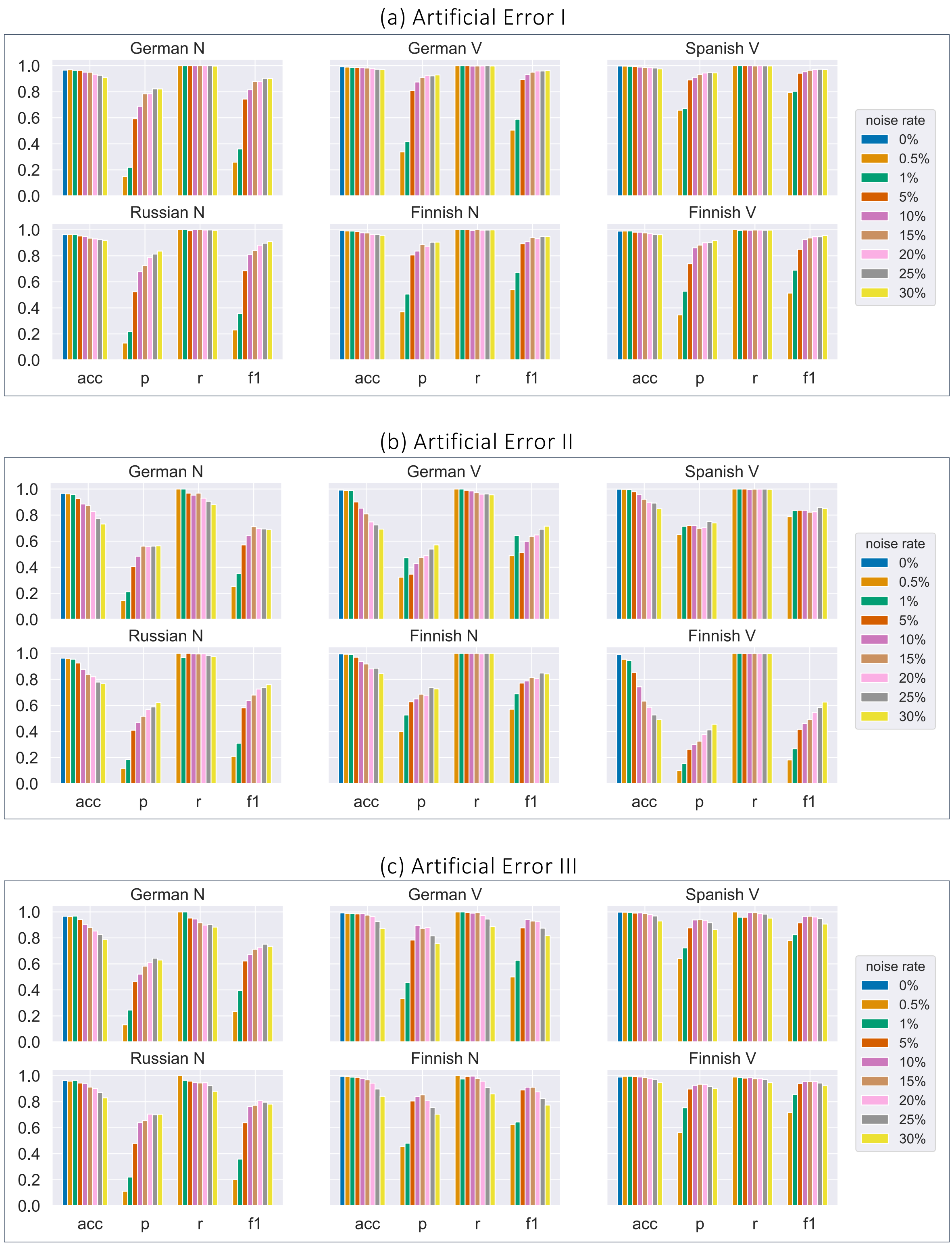
Table 1: Basic data information. The last column presents the Transformer inflection model performance (average accuracy) when no artificial error is inserted.

2.2 Experiment setup

Inflection model The Transformer (Vaswani et al., 2017) is the current state-of-the-art model architecture for morphological inflection generation, even when the amount of training data is limited (Vylomova et al., 2020; Liu and Hulden, 2020a,b, 2021a,b; Moeller et al., 2020, 2021; Wu et al., 2021; Liu, 2021); we therefore adopt this architecure in all experiments.[1]

Applying the Transformer to detect morphological data errors The core intuition behind our error detection model is that we *train inflection generation models* on a subset of the inflected forms in our total dataset, and then *apply these models*

[1] We implement all models in FAIRSEQ (Ott et al., 2019) and the hyperparameter setting follow Liu and Hulden (2020a) exactly.


(a) Artificial Error I

(b) Artificial Error II

(c) Artificial Error III

Figure 2: Model performance on adding different types of artificial errors. In each group, the bars from left to right show results for introducing an increasingly larger amount of artificial errors. Accuracy (acc) is the inflection model performance. Precision (p), recall (r) and F1-score (f_1) evaluate the effectiveness of error detection with the inflection model. p, r and f_1 are not applicable when no artificial error, i.e. 0%, is introduced.

to generate precisely those inflected forms that the inflection models have *not* *been trained on*. If a model's prediction for these forms disagrees with the corresponding held-out annotated form, we flag that particular annotated form as a potential error.

Preliminary experiment and data split Throughout our experiments, we use complete inflection tables for our labeled data. Moreover, the dataset is a small subset of the UniMorph tables, ranging from 70 tables (Spanish verbs) to 240 (Russian nouns). The reason for limiting the data is twofold. First, we want to ensure that error detection is feasible with datasets significantly smaller than large projects such as UniMorph. Secondly, before our actual error detection experiment, we want to verify that the Transformer model is powerful enough to reconstruct, with high accuracy, single unseen (or potentially erroneous) forms in the data.

We use a *leave-n-out* cross-validation setup to split the data for training and evaluating the model before attempting to perform error detection. Specifically, as illustrated in Figure 1, we systematically leave one slot out in each inflection table for evaluation and use the remaining slots to train one particular inflection model. For each such model, we rotate which slot is left out. The number of models we train for each POS of a language is thus the same as the corresponding paradigm size, m. The evaluation data size for each model is n, the same as the number inflection tables in the data, and the training data size for each model is $m \times n - n$. Each model is thus trained to make predictions for slots it has not witnessed—one missing slot per table—and the union of all models' predictions cover all the slots. Table 1 shows the accuracy when using the m models to perform an artificial reconstruction of "unseen forms". For example, we train $m = 8$ inflection models for German nouns, each model is trained on 1,120 ($8 \times 160 - 160$) slots and evaluated on $n = 160$ slots.

Generating artificial errors We now simulate noisy annotation data by injecting artificial errors into the above dataset in three different ways before training models. The first method generates artificial errors (Artificial Error I) to mimic typographic errors by *inserting, replacing or deleting* a single character in an inflected word form. The second error model simulates annotator confusion by *swapping two randomly sampled slots* with different inflected forms in a randomly chosen inflection table, denoted as Artificial Error II. The third type of artificial error, Artificial Error III, is self-adversarial to generate plausible-looking noise: we first train a single Transformer inflection model with the complete data for each POS of a language, then apply it to predict inflected forms for slots it *has* been trained on. We use beam search at decoding time and *pick out the second best (but erroneous) prediction* to represent a noisy inflected form. This self-adversarial approach gives us incorrect word forms which are however very close to the ground truth inflected word forms. We hypothesize that such errors are more difficult to identify than the others.

Erroneous inflected forms of each type are introduced to the original data at different error rates: 0.5%, 1%, 5%, 15%, 20%, 25% and 30% (of all forms).

Evaluation metrics We evaluate the error detection model w.r.t. *accuracy*, i.e. the ratio of correctly predicted forms vs. all predicted forms and also *precision*, *recall*, and *F1-score*.

3 Results and Discussion

Figure 2 provides a summary of the experiment results, plotting the accuracy, precision, recall, and F1-score for each POS of each language, averaged across the m models after adding Artificial Errors I, II, III at different amounts, respectively. Detailed numbers are provided in Table 2 in the appendix.

We observe that the accuracy of the model decreases as more word erroneous forms are added, but is still high overall. This indicates that the *leave-n-out* training strategy is robust to noise in the data. For every type of artificial error, the recall is 1.0 or very close to 1.0 after varying amounts of noise is injected. In other words, the model can identify all, or nearly all the artificial errors we introduce, even when a large amount of noise is mixed into the gold data. The precision increases (from a low of 0.11 to a high of 0.95) as more errors are added, indicating that a reasonably small amount of false positives would be produced by the model. (See Table 3 in the appendix for detailed counts.)

As such, if an annotator were to manually correct the forms flagged by the model, all erroneous annotations would be corrected and the annotator should not be frustrated by vetting a large number of already-correct annotations. To illustrate this, consider the average precision (0.43) for all six datasets with Artificial Error type I (typos) where 1% of the forms are corrupted—a plausible scenario in an annotation project. Under such assumptions, our model would present flagged forms in a dataset for vetting to an annotator, and, indeed, nearly half of these flagged forms would be true errors, and no errors would be undetected (since the recall is 1.0).

However, we observe that the worst case (e.g. lowest F1 scores on average) where the annotation error detection model performs is the second type of artificial error. In this type of error, we consistently switched a portion of slots. The worst error detection model performance on this type of error points to the limitation of the annotation error detection method we propose: it cannot detect consistent errors if the errors in question are present in a large portion of the data; for example, in the extreme case that all the forms in the paradigm carry the same error, it is impossible for the inflection model to learn the ground-truth inflection. Another shortcoming of our proposed approach is that it requires relatively complete inflection tables, which are expensive to annotate as to expertise and effort. Future work is needed to evaluate whether the method works when there are slots missing in most inflection tables.

4 Conclusion

In this work, we propose a method to leverage the Transformer model architecture for annotation error detection in morphological data. We propose to systematically leave out one slot in each morphological inflection table as the data to be detected and use such subsets of annotated data to train individual Transformer inflection models—one for each group of missing slots—and then apply the inflection models to make predictions for the held-out slots. If the predicted form disagrees with the actual annotation (a form the predicting model has not seen), the model flags that form as erroneous.

To check efficiency, we evaluate the model under three different scenarios where we inject artificial errors into gold data, simulating noisy data resulting from an annotation process: typographic errors generated by inserting, replacing or deleting a single character in an inflected word form; errors resulting from annotator confusion where two slots in an inflection table are swapped; and self-adversarial errors where erroneous but plausible predictions generated by the Transformer inflection model are introduced. Our experiments on four languages with different morphological characteristics and levels of irregularity indicate that the proposed method can detect every type of error in morphological datasets very effectively. Even when large portions of the data (5% to 30%) have been replaced with corrupted forms, our model retains perfect, or near-perfect, recall and also shows increasingly higher precision as more erroneous forms are present.

The results show that the Transformer model can detect various kinds of errors without producing excessive false positive predictions. We believe such a model can directly be incorporated into the correction and quality control process of morphological data annotation projects, specifically for low-resource language where datasets are in the early stages of development and few annotators are available. Further research should investigate how well this basic method of error detection works in other linguistic annotation domains.

References

Bharat Ram Ambati, Rahul Agarwal, Mridul Gupta, Samar Husain, and Dipti Misra Sharma. 2011. Error detection for treebank validation. In *Proceedings of the 9th Workshop on Asian Language Resources*, pages 23–30, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017. CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 1–30, Vancouver. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The SIGMORPHON 2016 shared Task—Morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22, Berlin, Germany. Association for Computational Linguistics.

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

Markus Dickinson and Chong Min Lee. 2008. Detecting errors in semantic annotation. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Markus Dickinson and W. Detmar Meurers. 2003. Detecting errors in part-of-speech annotation. In *10th Conference of the European Chapter of the Association for Computational Linguistics*, Budapest, Hungary. Association for Computational Linguistics.

Eleazar Eskin. 2000. Detecting errors within a corpus using anomaly detection. In *1st Meeting of the North American Chapter of the Association for Computational Linguistics*.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sabrina J. Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: Universal Morphology. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Pavel Květoň and Karel Oliva. 2002. (semi-)automatic detection of errors in PoS-tagged corpora. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Ling Liu. 2021. Computational morphology with neural network approaches. *arXiv preprint arXiv:2105.09404*.

Ling Liu and Mans Hulden. 2020a. Analogy models for neural word inflection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2861–2878, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ling Liu and Mans Hulden. 2020b. Leveraging principal parts for morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 153–161, Online. Association for Computational Linguistics.

Ling Liu and Mans Hulden. 2021a. Backtranslation in neural morphological inflection. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 81–88, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ling Liu and Mans Hulden. 2021b. Can a transformer pass the wug test? Tuning copying bias in neural morphological inflection models. *arXiv preprint arXiv:2104.06483*.

Hrafn Loftsson. 2009. Correcting a POS-tagged corpus using three complementary methods. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 523–531, Athens, Greece. Association for Computational Linguistics.

Arya D. McCarthy, Christo Kirov, Matteo Grella, Amrit Nidhi, Patrick Xia, Kyle Gorman, Ekaterina Vylomova, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, Timofey Arkhangelskiy, Nataly Krizhanovsky, Andrew Krizhanovsky, Elena Klyachko, Alexey Sorokin, John Mansfield, Valts Ernštreits, Yuval Pinter, Cassandra L. Jacobs, Ryan Cotterell, Mans Hulden, and David Yarowsky. 2020. UniMorph 3.0: Universal Morphology. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3922–3931, Marseille, France. European Language Resources Association.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sabrina J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy. Association for Computational Linguistics.

Sarah Moeller, Ling Liu, and Mans Hulden. 2021. To POS tag or not to POS tag: The impact of POS tags on morphological learning in low-resource settings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 966–978, Online. Association for Computational Linguistics.

Sarah Moeller, Ling Liu, Changbing Yang, Katharina Kann, and Mans Hulden. 2020. IGT2P: From interlinear glossed texts to paradigms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5251–5262, Online. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. FAIRSEQ: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Ines Rehbein and Josef Ruppenhofer. 2017. Detecting annotation noise in automatically labelled data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1160–1170, Vancouver, Canada. Association for Computational Linguistics.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99, Vancouver. Association for Computational Linguistics.

Hans van Halteren. 2000. The detection of inconsistency in manually tagged text. In *Proceedings of the COLING-2000 Workshop on Linguistically Interpreted Corpora*, pages 48–55, Centre Universitaire, Luxembourg. International Committee on Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Ekaterina Vylomova, Jennifer White, Elizabeth Salesky, Sabrina J. Mielke, Shijie Wu, Edoardo Maria Ponti, Rowan Hall Maudslay, Ran Zmigrod, Josef Valvoda, Svetlana Toldova, Francis Tyers, Elena Klyachko, Ilya Yegorov, Natalia Krizhanovsky, Paula Czarnowska, Irene Nikkarinen, Andrew Krizhanovsky, Tiago Pimentel, Lucas Torroba Hennigen, Christo Kirov, Garrett Nicolai, Adina Williams, Antonios Anastasopoulos, Hilaria Cruz, Eleanor Chodroff, Ryan Cotterell, Miikka Silfverberg, and Mans Hulden. 2020. SIGMORPHON 2020 shared task 0: Typologically diverse morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 1–39, Online. Association for Computational Linguistics.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2021. Applying the transformer to character-level transduction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1901–1907, Online. Association for Computational Linguistics.

A Detailed experiment results

	Artificial	**Artificial Error I**				**Artificial Error II**				**Artificial Error III**			
	Error Rate	acc	p	r	f1	acc	p	r	f1	acc	p	r	f1
German N	0%	0.9664	N/A	N/A	N/A	0.9664	N/A	N/A	N/A	0.9664	N/A	N/A	N/A
	0.5%	0.9688	0.1489	1.0	0.2592	0.9625	0.1455	1.0	0.254	0.9641	0.1321	1.0	0.2334
	1%	0.9641	0.2203	1.0	0.3611	0.9586	0.2121	1.0	0.35	0.968	0.2453	1.0	0.394
	5%	0.9641	0.5926	1.0	0.7442	0.9258	0.4052	1.0	0.5714	0.9422	0.4621	0.9531	0.6224
	10%	0.9508	0.6882	1.0	0.8153	0.8852	0.4841	1.0	0.6421	0.9031	0.5216	0.9453	0.6723
	15%	0.95	0.7837	1.0	0.8787	0.8742	0.5619	1.0	0.7113	0.8789	0.5828	0.9167	0.7126
	20%	0.9344	0.7853	1.0	0.8797	0.8281	0.5574	1.0	0.6969	0.8531	0.6117	0.8984	0.7278
	25%	0.9266	0.8226	1.0	0.9027	0.7742	0.562	1.0	0.6938	0.8258	0.6437	0.9031	0.7516
	30%	0.9094	0.8219	0.9974	0.9012	0.732	0.5643	0.9974	0.6877	0.7883	0.6301	0.8828	0.7353
Russian N	0%	0.9625	N/A	N/A	N/A	0.9625	N/A	N/A	N/A	0.9625	N/A	N/A	N/A
	0.5%	0.9653	0.1304	1.0	0.2307	0.958	0.1168	1.0	0.2092	0.958	0.1111	1.0	0.2
	1%	0.9632	0.218	1.0	0.358	0.9549	0.1847	1.0	0.3101	0.9649	0.2205	0.9655	0.359
	5%	0.9524	0.5238	0.9931	0.6859	0.9253	0.4103	0.9931	0.5819	0.9438	0.4792	0.9583	0.6389
	10%	0.9483	0.6776	1.0	0.8078	0.8771	0.469	1.0	0.6378	0.9378	0.6393	0.9479	0.7636
	15%	0.9358	0.7248	1.0	0.8404	0.8378	0.5162	1.0	0.6798	0.9128	0.6549	0.9444	0.7734
	20%	0.9302	0.7888	0.9983	0.8813	0.8201	0.5706	0.9983	0.7257	0.9028	0.705	0.9462	0.808
	25%	0.924	0.8133	0.9986	0.8965	0.7792	0.5884	0.9986	0.7366	0.8722	0.6993	0.9236	0.7959
	30%	0.9194	0.8367	0.9965	0.9096	0.766	0.622	0.9965	0.759	0.8302	0.7031	0.8796	0.7815
Finnish N	0%	0.9959	N/A	N/A	N/A	0.9959	N/A	N/A	N/A	0.9959	N/A	N/A	N/A
	0.5%	0.9913	0.3704	1.0	0.5406	0.9923	0.4	1.0	0.5714	0.9939	0.4545	1.0	0.625
	1%	0.9901	0.5063	1.0	0.6722	0.9908	0.5263	1.0	0.6896	0.989	0.4815	0.975	0.6446
	5%	0.9862	0.8066	1.0	0.8929	0.9702	0.6282	1.0	0.7716	0.9875	0.8058	0.9949	0.8904
	10%	0.977	0.8369	0.9949	0.9091	0.9378	0.6501	0.9949	0.788	0.9788	0.8391	0.9974	0.9114
	15%	0.976	0.8855	1.0	0.9393	0.9184	0.6861	1.0	0.8138	0.9681	0.8531	0.9779	0.9112
	20%	0.9643	0.8737	0.9974	0.9315	0.8804	0.678	0.9974	0.8069	0.9429	0.8075	0.9579	0.8763
	25%	0.9638	0.9047	0.998	0.9491	0.8852	0.7368	0.998	0.8485	0.8982	0.7557	0.9092	0.8254
	30%	0.9571	0.9052	0.9991	0.9498	0.8434	0.7273	0.9991	0.8421	0.8418	0.7042	0.8605	0.7745
German V	0%	0.9919	N/A	N/A	N/A	0.9919	N/A	N/A	N/A	0.9919	N/A	N/A	N/A
	0.5%	0.9895	0.3385	1.0	0.5058	0.9891	0.3235	1.0	0.4889	0.9895	0.3333	1.0	0.5
	1%	0.9857	0.4175	1.0	0.5891	0.9883	0.4731	1.0	0.6423	0.9879	0.4574	1.0	0.6277
	5%	0.9874	0.8084	1.0	0.894	0.9006	0.3471	1.0	0.5141	0.985	0.7836	0.9953	0.8769
	10%	0.9843	0.875	0.9976	0.9323	0.8528	0.4293	0.9976	0.5981	0.986	0.8968	0.9905	0.9413
	15%	0.9826	0.9078	0.9984	0.9509	0.8098	0.4749	0.9984	0.6379	0.9753	0.8733	0.9937	0.9296
	20%	0.9793	0.9231	0.9988	0.9595	0.7477	0.4888	0.9988	0.648	0.9636	0.8797	0.9738	0.9244
	25%	0.9729	0.922	1.0	0.9594	0.7244	0.5397	1.0	0.6915	0.9272	0.8154	0.9449	0.8754
	30%	0.9693	0.9292	0.9984	0.9626	0.6923	0.5716	0.9984	0.7153	0.8728	0.7571	0.8867	0.8168
Spanish V	0%	0.998	N/A	N/A	N/A	0.998	N/A	N/A	N/A	0.998	N/A	N/A	N/A
	0.5%	0.9973	0.6579	1.0	0.7937	0.9971	0.65	1.0	0.7879	0.9971	0.641	1.0	0.7812
	1%	0.9951	0.6712	1.0	0.8033	0.9959	0.7143	1.0	0.8333	0.9959	0.7231	0.9592	0.8246
	5%	0.9937	0.8909	1.0	0.9423	0.9794	0.7193	1.0	0.8367	0.9908	0.8769	0.9592	0.9162
	10%	0.9894	0.9108	1.0	0.9533	0.9573	0.7208	1.0	0.8363	0.992	0.9383	0.9939	0.9653
	15%	0.9873	0.9327	0.9986	0.9645	0.921	0.698	0.9986	0.8217	0.9884	0.9396	0.9946	0.9663
	20%	0.9849	0.9441	1.0	0.9712	0.898	0.7033	1.0	0.8255	0.98	0.9353	0.9878	0.9608
	25%	0.9829	0.9481	0.9992	0.973	0.8924	0.752	0.9992	0.8582	0.9688	0.917	0.9829	0.9488
	30%	0.9753	0.9453	0.9986	0.9712	0.8484	0.74	0.9986	0.8496	0.93	0.8653	0.9524	0.9068
Finnish V	0%	0.9896	N/A	N/A	N/A	0.9896	N/A	N/A	N/A	0.9896	N/A	N/A	N/A
	0.5%	0.9905	0.346	1.0	0.5141	0.9545	0.1003	1.0	0.1823	0.9961	0.5625	0.99	0.7174
	1%	0.991	0.528	0.995	0.6899	0.9442	0.1542	0.995	0.2672	0.9966	0.7538	0.9849	0.854
	5%	0.9818	0.7394	0.998	0.8495	0.8527	0.2631	0.998	0.4164	0.9934	0.898	0.9819	0.9381
	10%	0.9818	0.8618	0.997	0.9245	0.743	0.3007	0.997	0.4622	0.9902	0.9257	0.9839	0.9539
	15%	0.9765	0.8826	0.9983	0.9369	0.6335	0.3259	0.9983	0.4914	0.9855	0.9349	0.9769	0.9554
	20%	0.971	0.9002	0.998	0.9466	0.5865	0.3759	0.998	0.5463	0.9805	0.9317	0.9806	0.9555
	25%	0.9633	0.9006	0.997	0.9464	0.5266	0.4121	0.997	0.5832	0.9688	0.9182	0.971	0.9439
	30%	0.9622	0.9178	0.9977	0.9561	0.4915	0.4559	0.9977	0.6258	0.9493	0.9003	0.9465	0.9228

Table 2: Model performance in details on adding artificial errors of different types in different amounts. This is the information used to create Figure 2 in section 3. When no artificial errors, i.e. 0%, are introduced, precision, recall and F1-score are not applicable.

	Artificial	**Artificial Error I**			**Artificial Error II**			**Artificial Error III**		
	Error	True	Detected	Artificial	True	Detected	Artificial	True	Detected	Artificial
	Rate	Positive	Error	Error	Positive	Error	Error	Positive	Error	Error
German N	0%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	0.5%	7	47	7	8	47	8	7	53	7
	1%	13	59	13	14	59	14	13	53	13
	5%	64	108	64	62	108	64	61	132	64
	10%	128	186	128	122	186	128	121	232	128
	15%	192	245	192	186	245	192	176	302	192
	20%	256	326	256	238	326	256	230	376	256
	25%	320	389	320	290	389	320	289	449	320
	30%	383	466	384	338	466	384	339	538	384
Russian N	0%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	0.5%	15	115	15	16	115	16	15	135	15
	1%	29	133	29	29	133	30	28	127	29
	5%	143	273	144	144	273	144	138	288	144
	10%	288	425	288	287	425	288	273	427	288
	15%	432	596	432	430	596	432	408	623	432
	20%	575	729	576	574	729	576	545	773	576
	25%	719	884	720	709	884	720	665	951	720
	30%	861	1029	864	841	1029	864	760	1081	864
Finnish N	0%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	0.5%	20	54	20	20	54	20	20	44	20
	1%	40	79	40	40	79	40	39	81	40
	5%	196	243	196	196	243	196	195	242	196
	10%	390	466	392	392	466	392	391	466	392
	15%	588	664	588	588	664	588	575	674	588
	20%	782	895	784	781	895	784	751	930	784
	25%	978	1081	980	980	1081	980	891	1179	980
	30%	1175	1298	1176	1176	1298	1176	1012	1437	1176
German V	0%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	0.5%	22	65	22	22	65	22	22	66	22
	1%	43	103	43	44	103	44	43	94	43
	5%	211	261	211	210	261	212	210	268	211
	10%	420	480	421	416	480	422	417	465	421
	15%	630	694	631	614	694	632	627	718	631
	20%	840	910	841	809	910	842	819	931	841
	25%	1052	1141	1052	1012	1141	1052	994	1219	1052
	30%	1260	1356	1262	1206	1356	1262	1119	1478	1262
Spanish V	0%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	0.5%	25	38	25	26	38	26	25	39	25
	1%	49	73	49	50	73	50	47	65	49
	5%	245	275	245	246	275	246	235	268	245
	10%	490	538	490	488	538	490	487	519	490
	15%	734	787	735	735	787	736	731	778	735
	20%	980	1038	980	979	1038	980	968	1035	980
	25%	1224	1291	1225	1225	1291	1226	1204	1313	1225
	30%	1468	1553	1470	1466	1553	1470	1400	1618	1470
Finnish V	0%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	0.5%	100	289	100	100	289	100	99	176	100
	1%	198	375	199	200	375	200	196	260	199
	5%	993	1343	995	994	1343	996	977	1088	995
	10%	1983	2301	1989	1987	2301	1990	1957	2114	1989
	15%	2978	3374	2983	2980	3374	2984	2914	3117	2983
	20%	3969	4409	3977	3975	4409	3978	3900	4186	3977
	25%	4956	5503	4971	4957	5503	4972	4827	5257	4971
	30%	5951	6484	5965	5951	6484	5966	5646	6271	5965

Table 3: Count of errors. “True Positive” column lists the count of errors which are artificial errors we introduce to the data and identified by the model as being erroneous. “Detected Error” column lists the number of inflected forms which the model detects as being erroneous, and the inflection model is trained with corrupted data by adding artificial errors at different amounts. “Artificial Error” column lists the number of artificial errors for each artificial error type we introduce to the original morphological data.

Estimating the Entropy of Linguistic Distributions

Aryaman Arora **Clara Meister**[⁉] **Ryan Cotterell**[⁉]
Georgetown University [⁉]ETH Zürich
aa2190@georgetown.edu {clara.meister,ryan.cotterell}@inf.ethz.ch

Abstract

Shannon entropy is often a quantity of interest to linguists studying the communicative capacity of human language. However, entropy must typically be estimated from observed data because researchers do not have access to the underlying probability distribution that gives rise to these data. While entropy estimation is a well-studied problem in other fields, there is not yet a comprehensive exploration of the efficacy of entropy estimators for use with *linguistic* data. In this work, we fill this void, studying the empirical effectiveness of different entropy estimators for linguistic distributions. In a replication of two recent information-theoretic linguistic studies, we find evidence that the reported effect size is over-estimated due to over-reliance on poor entropy estimators. Finally, we end our paper with concrete recommendations for entropy estimation depending on distribution type and data availability.

1 Introduction

There is a natural connection between information theory, the mathematical study of communication systems, and linguistics, the study of human language—the primary vehicle that humans employ to communicate. Researchers have exploited this connection since information theory's inception (Shannon, 1951; Cherry et al., 1953; Harris, 1991). With the advent of modern computing, the number of information-theoretic linguistic studies has risen, exploring claims about language such as the optimality of the lexicon (Piantadosi et al., 2011; Pimentel et al., 2021), the complexity of morphological systems (Cotterell et al., 2019; Wu et al., 2019; Rathi et al., 2021), and the correlation between surprisal and language processing time (Smith and Levy, 2013; Bentz et al., 2017; Goodkind and Bicknell, 2018; Cotterell et al., 2018; Meister et al., 2021, *inter alia*).

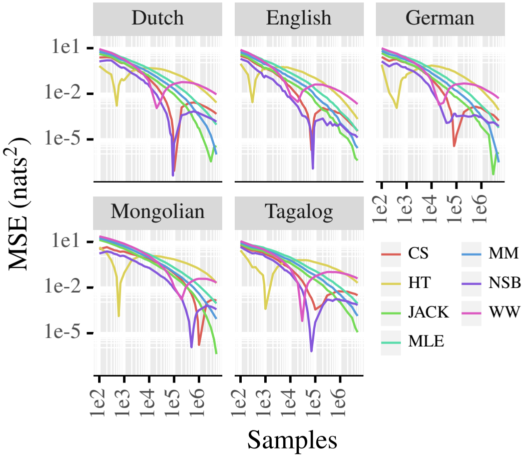


Figure 1: A comparison of several estimators of the entropy of the unigram distribution across 5 languages. Minima in all the graphs indicate sign changes in the error of the estimate, from an under- to an over-estimate.

In information-theoretic linguistics, a fundamental quantity of research interest is entropy. Entropy is both useful to linguists in its own right, and is necessary for estimating other useful quantities, e.g., mutual information. However, the estimation of entropy from raw data can be quite challenging (Paninski, 2003; Nowozin, 2015), e.g., in expectation, the plug-in estimator *underestimates* entropy (Miller, 1955). Linguistic distributions often present additional challenges. For instance, many linguistic distributions, such as the unigram distribution, follow a power law (Zipf, 1935; Mitzenmacher, 2004).[1] Linguistics is not the only field with such nuances, and so a large number of entropy estimators have been proposed in other fields (Chao and Shen, 2003; Archer et al., 2014, *inter alia*). However, no work to date has attempted a practical comparison of these estimators on *natural language* data. This work fills this empirical void.

Our paper offers a large empirical comparison of the performance of 6 different entropy estimators

[1] As Nemenman et al. (2002) highlight, when estimating the entropy of a distribution that follows a power law, it is often possible to get an effectively meaningless estimate that is completely determined by the estimator's hyperparameters.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 175 - 195
May 22-27, 2022 ©2022 Association for Computational Linguistics

on both synthetic and natural language data, an example of which is shown in Figure 1. We find that Chao and Shen's (2003) is the best estimator when very few data are available, but Nemenman et al.'s (2002) is superior as more data become available. Both are significantly better (in terms of mean-squared error) than the naïve plug-in estimator. Importantly, we also show that two recent studies (Williams et al., 2021; McCarthy et al., 2020) show smaller effect sizes when a better estimator is employed; however, we are able to reproduce a significant effect in both replications. We recommend that future studies carefully consider their choice of entropy estimators, taking into account data availability and the nature of the underlying distribution.[2]

2 Entropy and Language

Shannon entropy is a quantification of the uncertainty in a random variable. Given a (discrete) random variable X with probability distribution p over K possible outcomes $\mathcal{X} = \{x_k\}_{k=1}^{K}$, the Shannon entropy of X is defined as

$$\mathrm{H}(X) = \mathrm{H}(p) \stackrel{\text{def}}{=} -\sum_{k=1}^{K} p(x_k) \log p(x_k) \tag{1}$$

Entropy has many uses throughout science and engineering; for instance, Shannon (1948) originally proposed entropy as a lower bound on the compressibility of a stochastic source.

Yet the application of information-theoretic techniques to linguistics is not so straightforward: Information-theoretic measures are defined over probability distributions and, in the study of natural language, we typically only have access to *samples* from the distribution of interest, e.g., the phonotactic distribution in English, which permits word we cannot find in a corpus, like *blick*, rather than the true probabilities required in the computation of Eq. (1). Indeed, it is often the case that not all elements of $\mathcal{X}$ are even observed in available data—such as words that were coined after the a corpus was collected.

Rather, p must be approximated in order to estimate $\mathrm{H}(p)$. One solution is **plug-in estimation**: Given samples from p, the maximum-likelihood estimate for p is "plugged" into Eq. (1). However, as originally noted by Miller (1955), this strategy generally yields poor estimates.[3] It is thus necessary to derive more nuanced estimators.

3 Statistical Estimation Theory

Statistical estimation theory provides us with the tools for estimating various quantities of interest based on samples from a distribution.

Central to this theory is the **estimator**: A statistic that approximates a property of the distribution our data is drawn from. More formally, let $\mathcal{D} = \{\widetilde{x}^{(n)}\}_{n=1}^{N}$ be samples from an unknown distribution p. Suppose we are interested in a quantity θ that can be computed as a function of the distribution p. An estimator $\widehat{\theta}(\mathcal{D})$ for θ is then a function of the data $\mathcal{D}$ that provides an approximation of θ.

Two properties of an estimator are often of interest: **bias**—the difference between the true value of θ and the expected value of our estimator $\widehat{\theta}(\mathcal{D})$ under p—and **variance**—how much $\widehat{\theta}(\mathcal{D})$ fluctuates from sample set to sample set:

$$\mathrm{bias}(\widehat{\theta}(\mathcal{D})) \stackrel{\text{def}}{=} \mathbb{E}_p[\widehat{\theta}(\mathcal{D})] - \theta \tag{2}$$

$$\mathrm{var}(\widehat{\theta}(\mathcal{D})) \stackrel{\text{def}}{=} \mathbb{E}_p[(\widehat{\theta}(\mathcal{D}) - \mathbb{E}_p[\widehat{\theta}(\mathcal{D})])^2] \tag{3}$$

It is desirable to construct an estimator that has both low bias and low variance. However, the **bias–variance** trade-off tells us that we often have to pick one, and we should focus on a balance between the two. This trade-off is evinced through mean-squared error (MSE), a metric oft-employed for assessing estimator quality:

$$\mathrm{MSE}(\widehat{\theta}(\mathcal{D})) = \mathrm{bias}(\widehat{\theta}(\mathcal{D}))^2 + \mathrm{var}(\widehat{\theta}(\mathcal{D})) \tag{4}$$

To recognize the trade-oft note that, for any fixed MSE, a decrease in bias must be compensated with an increase in variance and vice versa. Indeed, it is important to recognize that there is typically no single estimator that is seen as "best." Different estimators balance the bias–variance trade-off differently, making their perceived quality specific to one's use-case. Importantly, the effectiveness of an estimator also depends on the domain of interest. Consequently, an empirical study of various entropy estimators, which this paper provides, is necessary in order to determine which entropy estimators are best suited for linguistic distributions.

3.1 Plug-in Estimation of Entropy

A simple, two-step approach for estimating entropy is **plug-in** estimation. In the first step, we compute the maximum-likelihood estimate for p from our

[2] Our code is available at `https://github.com/aryamanarora/entropy-estimation`.

[3] A proof of this result in given in full in Proposition 1.

dataset $\mathcal{D}$ as follows

$$\widehat{p}_{\text{MLE}}(x_k) \stackrel{\text{def}}{=} \frac{\sum_{n=1}^{N} \mathbb{1}\{\widetilde{x}^{(n)} = x_k\}}{N} \tag{5}$$

In the second step, we plug Eq. (5) into Eq. (1) directly, which results in the estimator $\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D})$. So why is this a bad idea? While our probability estimates themselves are unbiased, entropy is a concave function. Consequently, by Jensen's inequality, this estimator is, in expectation, a *lower bound* on the true entropy (see App. E.1 for proof). Moreover, when $N \ll K$, which is often the case in power-law distributed data, the estimate becomes quite unreliable (Nemenman et al., 2002).

3.2 An Ensemble of Entropy Estimators

MM—Miller (1955) and Madow (1948). The first innovation in entropy estimation known to the authors is a simple fix derived from a first-order Taylor expansion of MLE (described above). The Miller–Madow estimator only involves a simple additive correction, which is shown below:

$$\widehat{\mathrm{H}}_{\text{MM}}(\mathcal{D}) \stackrel{\text{def}}{=} \widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) + \frac{K-1}{2N} \tag{6}$$

where K is size of the support of $\mathcal{X}$. The Miller–Madow correction should seem intuitive in that we add $\frac{K-1}{2N} \geq 0$ to compensate for the negative bias of the estimator. A full derivation of the Miller–Madow estimator is given in Proposition 2.

JACK—Zahl (1977). Next we consider the jackknife, which is a common strategy used to correct for the bias of statistical estimators. In the case of entropy estimation, we can apply the jackknife out of the box to correct the bias inherent in the MLE estimator. Explicitly, this is done by averaging plug-in entropy estimates $\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D})$ albeit with the n^{th} sample from the data removed; we denote this held-out plug-in estimator as $\widehat{\mathrm{H}}^{\setminus n}_{\text{MLE}}(\mathcal{D})$. Averaging these "held-out" plug-in estimators results in the following simple entropy estimator

$$\widehat{\mathrm{H}}_{\text{JACK}}(\mathcal{D}) \stackrel{\text{def}}{=} N\,\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) - \frac{N-1}{N}\sum_{n=1}^{N} \widehat{\mathrm{H}}^{\setminus n}_{\text{MLE}}(\mathcal{D}) \tag{7}$$

Note that the jackknife is applicable to any estimator, not just $\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D})$, and, thus, can be combined with any of the other approaches mentioned.

HT—Horvitz and Thompson (1952). Horvitz–Thompson is a general scheme for building estimators that employs importance weighting in order to more efficiently estimate a function of a random variable. Importantly, this estimator gives us the ability to compensate for situations where the probability of an outcome is so low that it is often not observed in a sample, which is often the case for e.g., power-law distributions.

While a full exposition of HT estimators is outside of the scope of this work, in essence, we can divide the expected probability of a class by each class's estimated inclusion probability to compensate for such situations. Given the true probability of an outcome $p(x_k)$, the probability that it occurs at least once in a sample of size N is $1-(1-p(x_k))^N$. The HT estimator for entropy is then defined as

$$\widehat{\mathrm{H}}_{\text{HT}}(\mathcal{D}) \stackrel{\text{def}}{=} -\sum_{k=1}^{K} \frac{\widehat{p}_{\text{MLE}}(x_k)\log \widehat{p}_{\text{MLE}}(x_k)}{1-(1-\widehat{p}_{\text{MLE}}(x_k))^N} \tag{8}$$

using our MLE probability estimates $\widehat{p}_{\text{MLE}}(x_k)$.

CS—Chao and Shen (2003). Chao–Shen modifies HT by multiplying the MLE probability estimates by an estimate of sample coverage. Formally, let f_1 be the number of observed singletons[4] in sample; our sample coverage can be estimated as $\widehat{C} = 1 - \frac{f_1}{N}$. The CS estimator is then computed as:

$$\widehat{\mathrm{H}}_{\text{CS}}(\mathcal{D}) \stackrel{\text{def}}{=} -\sum_{k=1}^{K} \frac{\widehat{C}\cdot\widehat{p}_{\text{MLE}}(x_k)\log \widehat{C}\cdot\widehat{p}_{\text{MLE}}(x_k)}{1-(1-\widehat{C}\cdot\widehat{p}_{\text{MLE}}(x_k))^N} \tag{9}$$

In the case that $f_1 = N$, we set $f_1 = N-1$ to ensure the estimated entropy is not 0.

WW—Wolpert and Wolf (1995). One family of entropy estimators in information theory is based on Bayesian principles. The first of these was the Wolpert–Wolf estimator, which uses a Dirichlet prior (with concentration parameter α and a uniform base distribution). This Bayesian estimator has a clean, closed form:

$$\widehat{\mathrm{H}}_{\text{WW}}(\mathcal{D} \mid \boldsymbol{\alpha}) \stackrel{\text{def}}{=} \psi\left(\widetilde{A}+1\right) - \sum_{k=1}^{K} \frac{\widetilde{\alpha}_k}{\widetilde{A}}\psi(\widetilde{\alpha}_k+1) \tag{10}$$

where $\widetilde{\alpha}_k = c(x_k) + \alpha_k$ (for the histogram count $c(x_k)$ of class k in the sample; this is analogous to Laplace smoothing), $\widetilde{A} = \sum_{k=1}^{K} \widetilde{\alpha}_k$, and ψ is the digamma function. A full derivation of Eq. (10) is given in Proposition 3. Unfortunately, Eq. (10) is

[4] A singleton (*hapax legomenon*) is an outcome which is observed only once in the sample.

	MAB				**MSE**			
	10^2	10^3	10^4	10^5	10^2	10^3	10^4	10^5
English	HT	HT	NSB	NSB	HT	HT	NSB	NSB
German	HT	HT	NSB	CS	HT	HT	NSB	CS
Dutch	HT	HT	NSB	CS	HT	HT	NSB	CS
Mongolian	NSB	HT	NSB	NSB	NSB	HT	NSB	NSB
Tagalog	HT	HT	NSB	NSB	HT	HT	NSB	NSB

Table 1: The best unigram entropy estimators on the corpora studied, tested on various N averaged over 100 samples. All differences are statistically significant on the permutation test; lighter color indicates fewer statistically significant comparisons on the Tukey test. *Scale*: significantly better than 6 5 4 3 2 1 0 other estimators.

very dependent on the choice of $\boldsymbol{\alpha}$: For large K, $\boldsymbol{\alpha}$ almost completely determines the final entropy estimate, an observation first made by Nemenman et al. (2002) which motivated their improved estimator described below.

NSB—Nemenman et al. (2002). Nemenman et al. (NSB) attempt to alleviate the Wolpert–Wolf estimator's dependence on $\boldsymbol{\alpha}$. They take $\boldsymbol{\alpha} = \alpha \cdot \mathbf{1}$, enforcing that the Dirichlet prior is symmetric, and develop a hyperprior over α that results in a near-uniform distribution over entropy. The hyperprior is given by

$$p_{\text{NSB}}(\alpha) \overset{\text{def}}{=} \frac{K\psi_1(K\alpha+1) - \psi_1(\alpha+1)}{\log K} \tag{11}$$

where ψ_1 is the trigamma function. A full derivation of Eq. (11) is given in Proposition 4. This choice of hyperprior mitigates the effect that the chosen α has on the entropy estimate. Nemenman et al.'s (2002) entropy estimator is then the posterior mean of the Wolpert–Wolft estimator taken under p_{NSB}:

$$\widehat{\mathrm{H}}_{\text{NSB}}(\mathcal{D}) = \int_0^\infty \widehat{\mathrm{H}}_{\text{WW}}(\mathcal{D} \mid \alpha \cdot \mathbf{1})\, p_{\text{NSB}}(\alpha)\, \mathrm{d}\alpha \tag{12}$$

Typically, numerical integration is used to quickly compute the unidimensional integral.

4 Experiments

Here we provide an evaluation of the entropy estimators presented in §3.2 on linguistic data.

4.1 Entropy of the Unigram Distribution

We start our study with a controlled experiment where we estimate the entropy of the truncated unigram distribution, the (finite) distribution over the frequent word tokens in a language without regard to context (Baayen et al., 2016; Diessel, 2017; Divjak, 2019; Nikkarinen et al., 2021). We renormalize the frequency counts of corpora in English, German, and Dutch (taken from CELEX; Baayen et al., 1995), as well as Mongolian and Tagalog (from Wikipedia[5]). We take this renormalization as a gold standard distribution, since we cannot access the underlying unigram distribution. We then draw samples of varying sizes ($N \in \{10^2, 10^3, 10^4, 10^5\}$) from the distribution of renormalized frequency counts to test the estimators' ability to recover the underlying distributions' entropy. While the renormalized frequency counts are not necessarily representative of the *true* unigram distribution, they nevertheless provide us with a controlled setting to benchmark various entropy estimators.

We evaluate the estimators on both bias and MSE, as defined in (2) and (4), as well as mean absolute bias (MAB). To test the statistical significance of differences in metrics between entropy estimators, we use paired permutation tests (Good, 2000) (sampling $1,000$ permutations) between pairs of estimators, checking MAB and MSE. We run Tukey's test (1949) to judge the statistical significance of differences in MAB and MSE between all pairs of estimators, which found only a few insignificant comparisons when N was large.

Results are shown in Table 1 and Figure 1. We find that NSB (followed closely by CS) converges almost to the true entropy from below using with only a few samples. HT is the best estimator for $N < 2,000$, but as N increases it tends to overestimate entropy to the point where its bias is greater than that of MLE. Besides HT, all estimators at all tested sample sizes N have lower MAB and MSE than MLE.

4.2 Replication of Williams et al. (2021)

Next, we turn to a replication of Williams et al.'s (2021) information-theoretic study on the associa-

[5] We used dumps from November 1, 2021: Mongolian and Tagalog; the extracted counts are available in our repository.

Language	n	**MLE**	**CS**	**MM**	**JACK**	**WW**	**NSB**
Italian	16, 856	🟩 20.00%	🟩 15.56%	🟩 16.43%	🟩 14.09%	🟩 19.67%	🟩 11.41%
Polish	15, 525	30.52%	23.48%	25.49%	🟩 21.75%	34.68%	🟩 17.07%
Portuguese	7, 409	🟩 27.60%	🟩 20.76%	🟩 22.51%	🟩 18.81%	🟩 33.32%	🟩 14.18%
Spanish	21, 408	🟩 20.50%	🟩 15.17%	🟩 16.44%	🟩 13.80%	🟩 21.04%	🟩 10.50%
Arabic	2, 483	45.31%	🟩 38.49%	40.99%	🟩 37.93%	🟩 49.09%	🟩 34.82%
Croatian	13, 856	31.35%	26.04%	26.62%	23.08%	35.66%	19.06%
Greek	3, 305	41.58%	33.17%	36.39%	32.32%	48.80%	27.00%

Table 2: Normalized mutual information, calculated with several estimators, between adjectives and the inanimate nouns they modify based on UD corpora. Colored-in cell means statistically significant NMI value.

tion between gendered inanimate nouns and their modifying adjectives. They estimate mutual information by using its familiar decomposition as the difference of two entropies: $\mathrm{MI}(X;Y) = \mathrm{H}(X) - \mathrm{H}(X \mid Y)$. The entropies $\mathrm{H}(X)$ and $\mathrm{H}(X \mid Y)$ are estimated independently and then their difference is computed. We replicate Williams et al.'s (2021) experiments using gold-parsed Universal Dependencies corpora, filtering out animate nouns with Multilingual WordNet (Bond and Foster, 2013). We rerun their experimental set-up using our full suite of entropy estimators to determine whether the relationship they posit remains significant, checking 3 more languages not in the original study.

We report results for normalized mutual information (dividing MI by maximum possible MI) in Table 2. We find that using NSB (the estimator we found most effective in §4.1) instead of MLE, nearly halves the measured effect in all languages. However, the effect remains statistically significant in 5 of 7 languages tested, including the 4 that were also in the original study.

4.3 Replication of McCarthy et al. (2020)

Finally, we turn our attention to McCarthy et al.'s (2020) study on the similarity between grammatical gender partitions between languages. Using information-theoretic measures, they found that closely related languages have more similar gender groupings of core lexical items. We replicate their experiment on Swadesh lists (Swadesh, 1955) for 10 European languages with different estimators, and find that hierarchical clustering over both mutual (MI) and variational information (VI) produces the same trees as the original study. In this case, using NSB, our recommended estimator, results in a reduced estimate of MI (e.g. Croatian–Slovak: 0.54 with MLE → 0.46 with NSB), but significance testing with 1,000 permutations finds the same pairs were statistically significant for both MI and VI regardless of estimator: all pairs of Slavic languages and Romance languages, and Bulgarian–Spanish (see Figure 2). Thus, we see a similar result here as in the previous replication.

5 Conclusion

This work presents the first empirical study comparing the performance of various entropy estimators for use with natural language distributions. From experiments on synthetic data (appendix) and natural data (CELEX), and two replication studies of recent papers in information-theoretic linguistics, we find that the oft-employed plug-in estimator of entropy can cause misleading results, e.g., the overestimates of effect sizes seen in both replication studies. The recommendation of our paper is that researchers should carefully consider their choice of entropy estimator based on data availability and the nature of the underlying distribution.

Ethics Statement

The authors foresee no ethical concerns with the research presented in this paper.

Acknowledgments

We thank Adina Williams, Lucas Torroba Hennigen, Tiago Pimentel, and the anonymous reviewers for feedback on the manuscript.

References

Evan Archer, Il Memming Park, and Jonathan W. Pillow. 2014. Bayesian entropy estimation for countable discrete distributions. *Journal of Machine Learning Research*, 15(81):2833–2868.

R. H. Baayen, R. Piepenbrock, and L. Gulikers. 1995. *CELEX2*. Linguistic Data Consortium, Philadelphia.

R. Harald Baayen, Petar Milin, and Michael Ramscar. 2016. Frequency in lexical processing. *Aphasiology*, 30(11):1174–1220.

Christian Bentz, Dimitrios Alikaniotis, Michael Cysouw, and Ramon Ferrer-i Cancho. 2017. The entropy of words—Learnability and expressivity across more than 1000 languages. *Entropy*, 19(6):275.

Francis Bond and Ryan Foster. 2013. Linking and extending an open multilingual Wordnet. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1352–1362, Sofia, Bulgaria. Association for Computational Linguistics.

Cristina Bosco, Simonetta Montemagni, and Maria Simi. 2013. Converting Italian treebanks: Towards an Italian Stanford dependency treebank. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 61–69, Sofia, Bulgaria. Association for Computational Linguistics.

Anne Chao and Tsung-Jen Shen. 2003. Nonparametric estimation of Shannon's index of diversity when there are unseen species in sample. *Environmental and Ecological Statistics*, 10(4):429–443.

E. Colin Cherry, Morris Halle, and Roman Jakobson. 1953. Toward the logical description of languages in their phonemic aspect. *Language*, pages 34–46.

Ryan Cotterell, Christo Kirov, Mans Hulden, and Jason Eisner. 2019. On the complexity and typology of inflectional morphological systems. *Transactions of the Association for Computational Linguistics*, 7:327–342.

Ryan Cotterell, Sabrina J. Mielke, Jason Eisner, and Brian Roark. 2018. Are all languages equally hard to language-model? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 536–541, New Orleans, Louisiana. Association for Computational Linguistics.

Holger Diessel. 2017. Usage-based linguistics. In *Oxford Research Encyclopedia of Linguistics*. Oxford University Press.

Dagmar Divjak. 2019. *Frequency in Language: Memory, Attention and Learning*. Cambridge University Press.

Herwig Friedl and Erwin Stampfer. 2002. Jackknife resampling. In *Encyclopedia of Environmetrics*, volume 2, pages 1089–1098. Wiley Chichester.

I. J. Good. 1953. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:237–264.

Phillip I. Good. 2000. *Permutation Tests: A Practical Guide to Resampling Methods for Testing Hypotheses*, 2$^{\text{nd}}$ edition. Springer.

Adam Goodkind and Klinton Bicknell. 2018. Predictive power of word surprisal for reading times is a linear function of language model quality. In *Proceedings of the 8th Workshop on Cognitive Modeling and Computational Linguistics (CMCL 2018)*, pages 10–18, Salt Lake City, Utah. Association for Computational Linguistics.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature*, 585(7825):357–362.

Zellig Harris. 1991. *A Theory of Language and Information: A Mathematical Approach*, 1 edition. Clarendon Press.

D. G. Horvitz and D. J. Thompson. 1952. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260):663–685.

William G. Madow. 1948. On the limiting distributions of estimates based on samples from finite universes. *The Annals of Mathematical Statistics*, pages 535–545.

Simone Marsili. 2016. simomarsili/ndd: Bayesian entropy estimation in Python - via the Nemenman-Schafee-Bialek algorithm.

Arya D. McCarthy, Adina Williams, Shijia Liu, David Yarowsky, and Ryan Cotterell. 2020. Measuring the similarity of grammatical gender systems by comparing partitions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5664–5675, Online. Association for Computational Linguistics.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal Dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria. Association for Computational Linguistics.

Clara Meister, Tiago Pimentel, Patrick Haller, Lena Jäger, Ryan Cotterell, and Roger Levy. 2021. Revisiting the Uniform Information Density hypothesis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 963–980, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

George Miller. 1955. Note on the bias of information estimates. In *Information Theory in Psychology: Problems and Methods*, pages 95–100. Free Press, Glencoe, IL.

Michael Mitzenmacher. 2004. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 1(2):226–251.

Ilya Nemenman, F. Shafee, and William Bialek. 2002. Entropy and inference, revisited. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

Irene Nikkarinen, Tiago Pimentel, Damián Blasi, and Ryan Cotterell. 2021. Modeling the unigram distribution. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3721–3729, Online. Association for Computational Linguistics.

Sebastian Nowozin. 2015. Estimating discrete entropy, part 1.

Liam Paninski. 2003. Estimation of entropy and mutual information. *Neural Computation*, 15:1191–1254.

Steven T. Piantadosi, Harry Tily, and Edward Gibson. 2011. Word lengths are optimized for efficient communication. *Proceedings of the National Academy of Sciences*, 108(9):3526–3529.

Tiago Pimentel, Irene Nikkarinen, Kyle Mahowald, Ryan Cotterell, and Damián Blasi. 2021. How (non-)optimal is the lexicon? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4426–4438, Online. Association for Computational Linguistics.

Prokopis Prokopidis, Elina Desipri, Maria Koutsombogera, Harris Papageorgiou, and Stelios Piperidis. 2005. Theoretical and practical issues in the construction of a Greek dependency treebank. In *Proceedings of the 4th Workshop on Treebanks and Linguistic Theories (TLT)*, pages 149–160.

Prokopis Prokopidis and Haris Papageorgiou. 2017. Universal Dependencies for Greek. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 102–106, Gothenburg, Sweden. Association for Computational Linguistics.

Alexandre Rademaker, Fabricio Chalub, Livy Real, Cláudia Freitas, Eckhard Bick, and Valeria de Paiva. 2017. Universal Dependencies for Portuguese. In *Proceedings of the Fourth International Conference on Dependency Linguistics (Depling 2017)*, pages 197–206, Pisa,Italy. Linköping University Electronic Press.

Neil Rathi, Michael Hahn, and Richard Futrell. 2021. An information-theoretic characterization of morphological fusion. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10115–10120, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Claude E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

Claude E. Shannon. 1951. Prediction and entropy of printed English. *Bell System Technical Journal*, 30(1):50–64.

Nathaniel J. Smith and Roger Levy. 2013. The effect of word predictability on reading time is logarithmic. *Cognition*, 128(3):302–319.

Otakar Smrž, Viktor Bielický, Iveta Kouřilová, Jakub Kráčmar, Jan Hajič, and Petr Zemánek. 2008. Prague Arabic Dependency Treebank: A word on the million words. In *Proceedings of the Workshop on Arabic and Local Languages*, pages 16–23, Marrakech, Morocco.

Morris Swadesh. 1955. Towards greater accuracy in lexicostatistic dating. *International Journal of American Linguistics*, 21(2):121–137.

Dima Taji, Nizar Habash, and Daniel Zeman. 2017. Universal Dependencies for Arabic. In *Proceedings of the Third Arabic Natural Language Processing Workshop*, pages 166–176, Valencia, Spain. Association for Computational Linguistics.

Mariona Taulé, M. Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel annotated corpora for Catalan and Spanish. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Sara Tonelli, Rodolfo Delmonte, and Antonella Bristot. 2008. Enriching the venice Italian treebank with dependency and grammatical relations. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

John Tukey. 1949. Comparing individual means in the analysis of variance. *Biometrics*, 5(2):99–114.

Tim Vieira. 2017. Estimating means in a finite universe.

Adina Williams, Ryan Cotterell, Lawrence Wolf-Sonkin, Damián Blasi, and Hanna Wallach. 2021. On the relationships between the grammatical genders of inanimate nouns and their co-occurring adjectives and verbs. *Transactions of the Association for Computational Linguistics*, 9:139–159.

David H. Wolpert and David R. Wolf. 1995. Estimating functions of probability distributions from a finite set of samples. *Physical Review E*, 52(6):6841.

Alina Wróblewska. 2018. Extended and enhanced Polish dependency bank in Universal Dependencies format. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 173–182, Brussels, Belgium. Association for Computational Linguistics.

Shijie Wu, Ryan Cotterell, and Timothy O'Donnell. 2019. Morphological irregularity correlates with frequency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5117–5126, Florence, Italy. Association for Computational Linguistics.

Samuel Zahl. 1977. Jackknifing an index of diversity. *Ecology*, 58(4):907–913.

Zhiyi Zhang. 2012. Entropy estimation in Turing's perspective. *Neural Computation*, 24(5):1368–1389.

George Kingsley Zipf. 1935. *The Psycho-Biology of Language*. Houghton-Mifflin, New York, NY, USA.

	MAB				MSE			
	10^1	10^2	10^3	10^4	10^1	10^2	10^3	10^4
2	HT	WW	WW	WW	WW	WW	WW	JACK
5	MM	WW	WW	JACK	MM	WW	WW	MM
10	JACK	CS	WW	MM	JACK	WW	WW	MLE
100	CS	CS	JACK	WW	CS	JACK	JACK	WW
1000	CS	HT	CS	JACK	CS	HT	CS	JACK

Table 3: Estimators with least MAB (mean absolute bias) and MSE (mean squared error) for various combinations of N and K sampling from **symmetric Dirichlet**. The lighter the color the fewer estimators the best estimator was found to be statistically significantly better than.

	MAB				MSE			
	10^1	10^2	10^3	10^4	10^1	10^2	10^3	10^4
100	CS	CS	CS	J	CS	CS	CS	J
1000	NSB	HT	NSB	J	CS	HT	NSB	J

Table 4: Estimators with least MAB (mean absolute bias) and MSE (mean squared error) for various combinations of N and K sampling from **Zipfian distributions**.

A Implementation

The code for each of the entropy estimators is implemented in Python using numpy (Harris et al., 2020), except for NSB which was taken from an existing efficient implementation in the ndd module (Marsili, 2016). We calculated entropies with base e (in nats).

B Experiments with simulated data

In our experiments with simulated data, we explore distributions sampled from a symmetric Dirichlet prior with varying number of classes K and known distributions of Zipfian form with various parameters. Words in natural languages have a roughly Zipfian distribution, with probability inversely proportional to rank (Zipf, 1935), and a symmetric Dirichlet distribution is analogous to e.g. POS tag label distributions in natural language. Thus, studying synthetic data from such distributions as a start is useful.

B.1 Experiment 1: Symmetric Dirichlet distributions

We sample $1,000$ distributions from a symmetric Dirichlet distribution with variable number of classes K, i.e. with paramater $\alpha = [\alpha_1, \ldots, \alpha_K] = [1, \ldots, 1]$. We calculate entropy estimates on different sample sizes N. Since we know the parameters of the true distribution, we can compare estimates with the true entropy. We do pairwise comparisons of the MAB and MSE of estimators, using paired permutation tests to establish significance. Table 3 shows our results, including significance tests. It is clear that when $N \gg K$, all of the estimators have nearly converged to the true value and estimator choice does not matter. However, in the low-sample regime some estimators are indeed significantly better at approximating the true entropy. Our results are mixed as to which estimator is best in what context; the one found to be most frequently significantly better than other estimators was Chao–Shen. What is clear is that MLE is never the best choice.

B.2 Experiment 2: Zipfian distributions

We sample $1,000$ finite Zipfian distributions with K classes which obey Zipf's law, that the probability of an outcome is inverse proportional to its rank. The experimental setup is the same as in Experiment 1. A Zipfian distribution approximates (but is not a perfect model of) the distribution of tokens in natural language text in some languages, including English, which was the basis for the law being proposed. Compare similar experiments on infinite Zipf distributions by Zhang (2012). Results are in Table 4.

C Replication of Williams et al. (2021)

We used the following UD treebanks:

- **Arabic**: PADT (Smrž et al., 2008; Taji et al., 2017);
- **Greek**: GDT (Prokopidis et al., 2005; Prokopidis and Papageorgiou, 2017);
- **Italian**: ISDT (Bosco et al., 2013), VIT (Tonelli et al., 2008);
- **Polish**: PDB (Wróblewska, 2018);
- **Portuguese**: GSD (McDonald et al., 2013), Bosque (Rademaker et al., 2017);
- **Spanish**: AnCora (Taulé et al., 2008), GSD (McDonald et al., 2013).

D Additional Figures

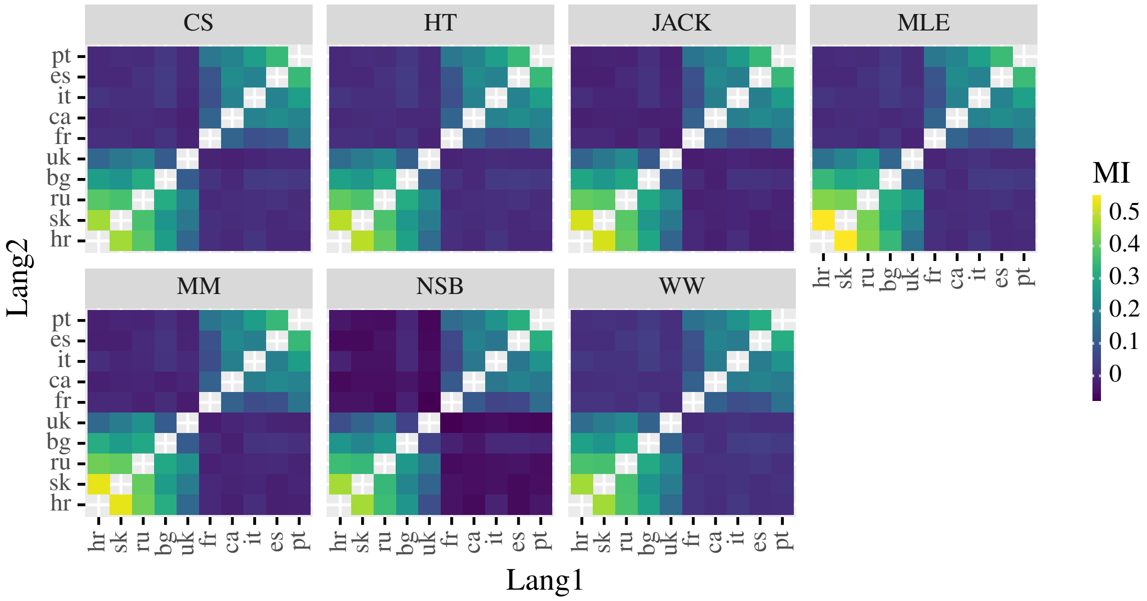


Figure 2: Mutual information between the gender partitions of language pairs with various estimators, replicating McCarthy et al. (2020).

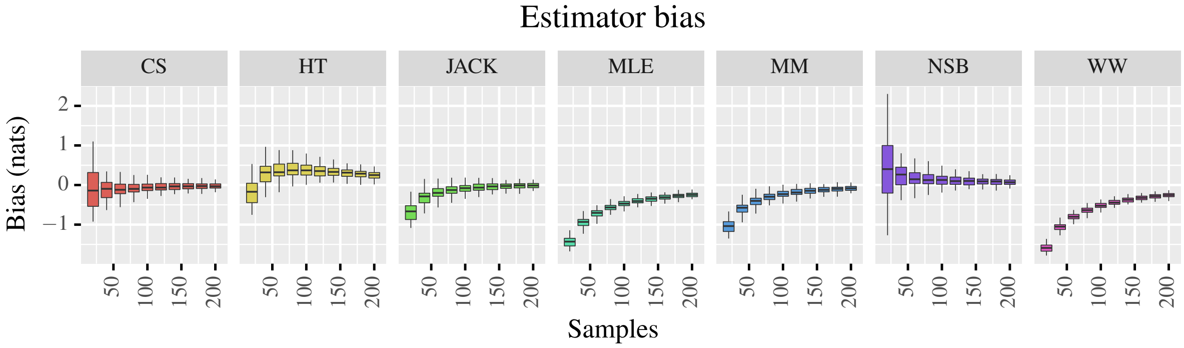
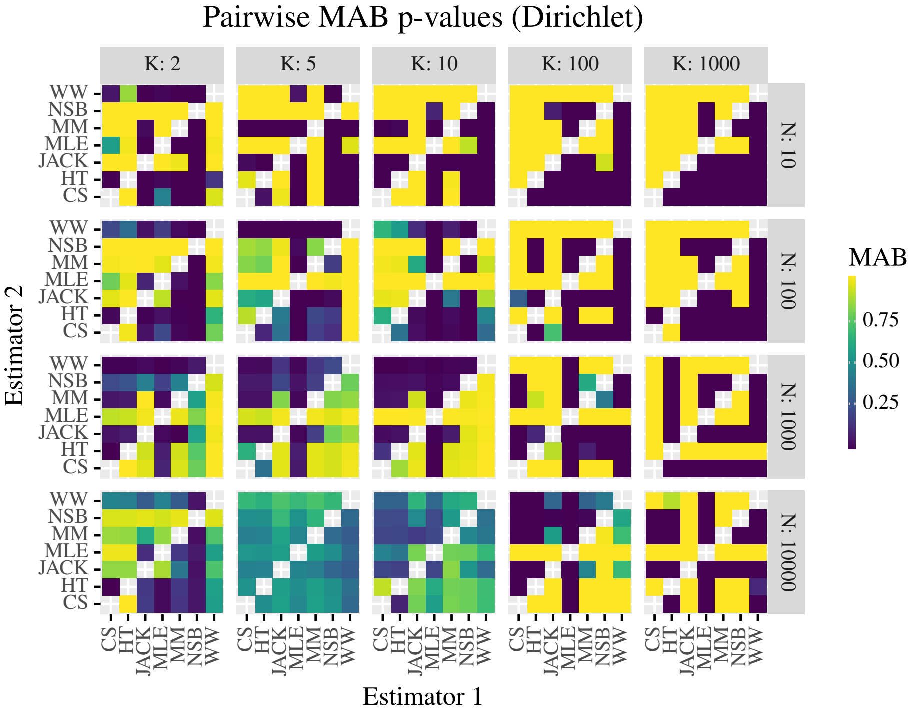


Figure 3: The distribution of bias for entropy over several estimators given variable sample size N, sampling from 100 distributions taken from a symmetric Dirichlet prior with $K = 100$.



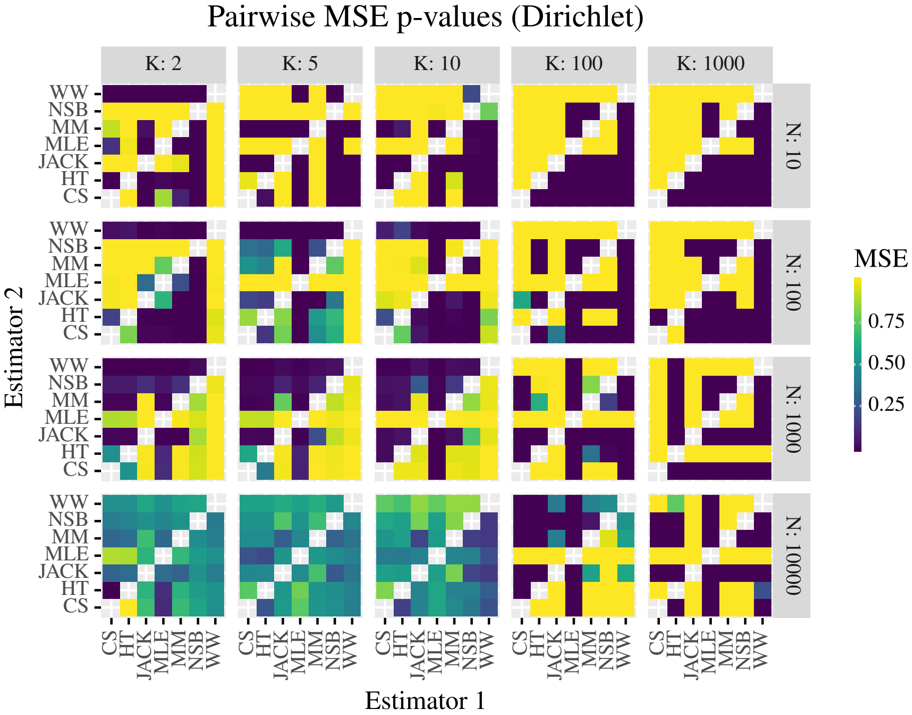


Figure 4: The heatmaps display the p-values calculated between pairs of estimators for mean absolute bias (MAB) and mean squared error (MSE) for Experiment 1. More purple values mean the estimator on the y-axis (Estimator 2) is better than the estimator on the x-axis (Estimator 1). Comparisons tend to become non-significant as N increases, since all the estimators gradually converge to the true entropy.

E Derivation of the Entropy Estimators

Let $\mathcal{X} = \{x_k\}_{k=1}^K$ be a finite set. Let p be a distribution over $\mathcal{X}$. The **entropy** of p is defined as

$$\mathrm{H}(p) \overset{\text{def}}{=} -\sum_{k=1}^{K} p_k \log p_k \tag{13}$$

Given a dataset of N samples $\mathcal{D}$ sampled i.i.d. from p, our goal is to estimate the entropy $\mathrm{H}(p)$ from samples $\mathcal{D}$ from the true distribution p. We will denote the count of an item x_k as $c(x_k) = \sum_{n=1}^N \mathbb{1}\left\{x_k = \widetilde{x}^{(n)}\right\}$. The **maximum-likelihood estimate** (MLE) of p given $\mathcal{D}$ is denoted $\frac{\sum_{n=1}^N \mathbb{1}\{\widetilde{x}^{(n)}=x_k\}}{N}$. The **plug-in estimate** of $\mathrm{H}(p)$ is defined to be the estimate of $\mathrm{H}(p)$ obtained by plugging the MLE estimate $\widehat{p}_{\text{MLE}}$ directly into the definition of entropy, i.e.,

$$\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) = \mathrm{H}(\widehat{p}_{\text{MLE}}) = -\sum_{k=1}^{K} \widehat{p}_{\text{MLE}}(x_k) \log \widehat{p}_{\text{MLE}}(x_k) = -\sum_{k=1}^{K} \frac{c(x_k)}{N} \log \frac{c(x_k)}{N} \tag{14}$$

This section discusses the problems with Eq. (14) as an estimator and provides detailed derivations of improved estimators found in the literature.

E.1 The Plug-in Estimator is Negatively Biased

Proposition 1. *The MLE entropy estimator in expectation underestimates true entropy, i.e.,*

$$\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) = \mathbb{E}\left[\sum_{k=1}^{K} -\widehat{p}_{\text{MLE}}(x_k) \log \widehat{p}_{\text{MLE}}(x_k)\right] \leq \mathrm{H}(p) \tag{15}$$

Proof. The result is a simple consequence of Jensen's inequality and some basic manipulations:

$$\begin{aligned}
\mathbb{E}\left[\sum_{k=1}^{K} -\widehat{p}_{\text{MLE}}(x_k) \log \widehat{p}_{\text{MLE}}(x_k)\right] &= \sum_{k=1}^{K} \mathbb{E}[-\widehat{p}_{\text{MLE}}(x_k) \log \widehat{p}_{\text{MLE}}(x_k)] && \text{(linearity of expectation)} \\
&\leq -\sum_{k=1}^{K} \mathbb{E}[\widehat{p}_{\text{MLE}}(x_k)] \log \mathbb{E}[\widehat{p}_{\text{MLE}}(x_k)] && \text{(Jensen's inequality)} \\
&= -\sum_{k=1}^{K} p(x_k) \log p(x_k) && (\mathbb{E}[\widehat{p}_{\text{MLE}}(x_k)] = p(x_k)) \\
&= \mathrm{H}(p) && \text{(definition of entropy)}
\end{aligned}$$

This completes the result. □

E.2 Miller–Madow

Proposition 2. *Let p be a categorical distribution over $\mathcal{X} = \{x_1, \ldots, x_K\}$, i.e., a categorical distribution with support K. Let $\mathcal{D}$ be our dataset of size N sampled from p. Finally, let $\widehat{p}_{\text{MLE}}$ be the maximum-likelihood estimate computed on $\mathcal{D}$. Then, we have*

$$\begin{aligned}
\text{bias}\left(\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D})\right) &\overset{\text{def}}{=} \mathbb{E}_p\left[\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D})\right] - \mathrm{H}(p) && (16) \\
&= -\frac{K-1}{2N} + o\left(N^{-1}\right) && (17)
\end{aligned}$$

Proof. We start by taking a first-order Taylor expansion and take an expectation of both sides.

$$\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) = \underbrace{\mathrm{H}(\widehat{p}_{\text{MLE}}, p)}_{\text{cross-entropy}} - \mathrm{KL}(\widehat{p}_{\text{MLE}} \mid\mid p) \qquad \text{(Lemma 1)} \tag{18}$$

$$\begin{aligned}
\mathbb{E}_p\left[\widehat{\mathrm{H}}_{\mathrm{MLE}}(\mathcal{D})\right] &= \mathbb{E}_p\left[\mathrm{H}(\widehat{p}_{\mathrm{MLE}}, p)\right] - \mathbb{E}_p\left[\mathrm{KL}(\widehat{p}_{\mathrm{MLE}} \mid\mid p)\right] && \text{(expectation)} && (19)\\
&= \mathbb{E}_p\left[-\sum_{k=1}^{K}\widehat{p}_{\mathrm{MLE}}(x_k)\log p(x_k)\right] - \mathbb{E}_p\left[\mathrm{KL}(\widehat{p}_{\mathrm{MLE}} \mid\mid p)\right] && \text{(defn. } \mathrm{H}(p,q)\text{)} && (20)\\
&= -\sum_{k=1}^{K}\mathbb{E}_p\left[\widehat{p}_{\mathrm{MLE}}(x_k)\log p(x_k)\right] - \mathbb{E}_p\left[\mathrm{KL}(\widehat{p}_{\mathrm{MLE}} \mid\mid p)\right] && \text{(linearity)} && (21)\\
&= -\sum_{k=1}^{K}\mathbb{E}_p\left[\widehat{p}_{\mathrm{MLE}}(x_k)\right]\log p(x_k) - \mathbb{E}_p\left[\mathrm{KL}(\widehat{p}_{\mathrm{MLE}} \mid\mid p)\right] && \text{(algebra)} && (22)\\
&= -\sum_{k=1}^{K} p(x_k)\log p(x_k) - \mathbb{E}_p\left[\mathrm{KL}(\widehat{p}_{\mathrm{MLE}} \mid\mid p)\right] && \text{(unbiased)} && (23)\\
&= \mathrm{H}(p) - \mathbb{E}_p\left[\mathrm{KL}(\widehat{p}_{\mathrm{MLE}} \mid\mid p)\right] && \text{(defn. of } \mathrm{H}(p)\text{)} && (24)\\
& && && (25)
\end{aligned}$$

This gives us:

$$\mathbb{E}_p\left[\widehat{\mathrm{H}}_{\mathrm{MLE}}(\mathcal{D})\right] - \mathrm{H}(p) = -\mathbb{E}_p\left[\mathrm{KL}(\widehat{p}_{\mathrm{MLE}} \mid\mid p)\right] \qquad \text{(subtract } \mathrm{H}(p)\text{)} \tag{26}$$

Thus, we may compactly write the bias as:

$$\begin{aligned}
\mathrm{bias}\left(\widehat{\mathrm{H}}_{\mathrm{MLE}}(\mathcal{D})\right) &= \mathbb{E}_p\left[\mathrm{H}(\widehat{p}_{\mathrm{MLE}})\right] - \mathrm{H}(p) && \text{(definition of bias)} && (27)\\
&= -\mathbb{E}_p\left[\mathrm{KL}(\widehat{p}_{\mathrm{MLE}} \mid\mid p)\right] && \text{(above computation)} && (28)\\
&\leq 0 && \text{(non-negativity of KL)} && (29)
\end{aligned}$$

Now, we find a simpler expression for the remainder $\mathbb{E}_p\left[\mathrm{KL}(\widehat{p}_{\mathrm{MLE}} \mid\mid p)\right]$. Again, we start with a second-order Taylor expansion

$$\mathrm{KL}(p \mid\mid q) = \sum_{x\in\mathcal{X}}\frac{\Delta(x)^2}{2q(x)} + o\left(\Delta(x)^2\right) \qquad \text{(Lemma 2)} \tag{30}$$

around the point $\Delta(x) = p(x) - q(x)$. Define $\widehat{p}_{\mathrm{MLE}}(x_k) = \frac{c(x_k)}{N}$ where $c(x_k)$ is the count of x_k in the training set. We now simplify the first term:

$$\begin{aligned}
\mathbb{E}_p\left[\sum_{k=1}^{K}\frac{\Delta(x_k)^2}{2q(x_k)}\right] &= \mathbb{E}_p\left[\sum_{k=1}^{K}\frac{(\widehat{p}_{\mathrm{MLE}}(x_k) - p(x_k))^2}{2p(x_k)}\right] && \text{(definition of } \Delta(x_k)\text{)} && (31)\\
&= \mathbb{E}_p\left[\sum_{k=1}^{K}\frac{(\frac{c(x_k)}{N} - p(x_k))^2}{2p(x_k)}\right] && \text{(definition of MLE)} && (32)\\
&= \mathbb{E}_p\left[\sum_{k=1}^{K}\frac{(c(x_k) - Np(x_k))^2}{2N^2p(x_k)}\right] && (\times {}^{N}/_{N}) && (33)\\
&= \frac{1}{2N}\mathbb{E}_p\left[\sum_{k=1}^{K}\frac{(c(x_k) - Np(x_k))^2}{Np(x_k)}\right] && \text{(pulling out } {}^{1}/_{2N}\text{)} && (34)\\
&= \frac{1}{2N}\mathbb{E}_p\left[\sum_{k=1}^{K}\frac{c(x_k)^2 - 2c(x_k)Np(x_k) + N^2p(x_k)^2}{Np(x_k)}\right] && \text{(exp. the binomial)} && (35)\\
&= \frac{1}{2N}\sum_{k=1}^{K}\frac{\mathbb{E}_p\left[c(x_k)^2\right] - 2Np(x_k)\mathbb{E}_p\left[c(x_k)\right] + N^2p(x_k)^2}{Np(x_k)} && \text{(lin. of expect.)} && (36)
\end{aligned}$$

$$= \frac{1}{2N}\sum_{k=1}^{K} \frac{\begin{array}{r}Np_k(1-p(x_k)) + N^2p(x_k)^2\\ -\,2N^2p(x_k)^2 + N^2p(x_k)^2\end{array}}{Np(x_k)} \qquad \text{(moments of MLE)} \tag{37}$$

$$= \frac{1}{2N}\sum_{k=1}^{K} \frac{Np_k(1-p(x_k))}{Np(x_k)} + \underbrace{\frac{1}{2N}\sum_{k=1}^{K} \frac{N^2p(x_k)^2 - 2N^2p(x_k)^2 + N^2p(x_k)^2}{Np(x_k)}}_{=0} \tag{38}$$

$$= \frac{1}{2N}\sum_{k=1}^{K} \frac{\cancel{Np(x_k)}(1-p(x_k))}{\cancel{Np(x_k)}} \tag{39}$$

$$= \frac{1}{2N}\sum_{k=1}^{K}(1-p(x_k)) \qquad \text{(algebra)} \tag{40}$$

$$= \frac{1}{2N}\underbrace{\sum_{k=1}^{K} 1}_{=K} - \frac{1}{2N}\underbrace{\sum_{k=1}^{K} p(x_k)}_{=1} \qquad \text{(algebra)} \tag{41}$$

$$= \frac{K-1}{2N} \tag{42}$$

Next, we simplify the second term, $o\left(\Delta(x)^2\right)$, in the MLE case:

$$\mathbb{E}_p\left[o\left(\Delta(x)^2\right)\right] = \mathbb{E}_p\left[o\left((\widehat{p}_{\text{MLE}}(x_k) - p(x_k))^2\right)\right] \qquad \text{(definition of } \Delta\text{)} \tag{43}$$

$$= \mathbb{E}_p\left[o\left(\left(\frac{c(x_k)}{N} - p(x_k)\right)^2\right)\right] \qquad \text{(definition of MLE)} \tag{44}$$

$$= \mathbb{E}_p\left[o\left(\frac{(c(x_k) - Np(x_k))^2}{N^2}\right)\right] \qquad (\times {}^{N}\!/{}_{N}) \tag{45}$$

$$= \mathbb{E}_p\left[o\left(\frac{c(x_k)^2 - 2c(x_k)Np(x_k) + N^2p(x_k)^2}{N^2}\right)\right] \tag{46}$$

$$= o\left(\frac{\mathbb{E}_p\left[c(x_k)^2 - 2c(x_k)Np(x_k) + N^2p(x_k)^2\right]}{N^2}\right) \qquad \text{(push exp. through)} \tag{47}$$

$$= o\left(\frac{\begin{array}{r}Np_k(1-p(x_k)) + N^2p(x_k)^2\\ -\,2N^2p(x_k)^2 + N^2p(x_k)^2\end{array}}{N^2}\right) \tag{48}$$

$$= o\left(\frac{Np(x_k)(1-p(x_k))}{N^2}\right) \qquad \text{(cancel terms)} \tag{49}$$

$$= o\left(\frac{p(x_k)(1-p(x_k))}{N}\right) \qquad \text{(cancel } N \text{ in fraction)} \tag{50}$$

$$= o\left(N^{-1}\right) \qquad \text{(ignore constants)} \tag{51}$$

Putting it all together, we get that $\text{bias}\left(\mathrm{H}(\widehat{p}_{\text{MLE}})\right) = -\frac{K-1}{2N} + o\left(N^{-1}\right)$ which is the desired result. □

Interestingly, it can be seen that the negative bias of the MLE gets worse as the number of classes K grows. Distributions with large K pop up frequently when dealing with natural language.

Corollary 1. *The plug-in estimator of entropy is consistent.*

Proof. From Proposition 2, we have $\text{bias}\left(\mathrm{H}(\widehat{p}_{\text{MLE}})\right) = -\frac{K-1}{2N} + o\left(N^{-1}\right)$. Clearly, as $N \to 0$, we have $\text{bias}\left(\mathrm{H}(\widehat{p}_{\text{MLE}})\right) \to 0$, so the estimator is consistent. One could also prove consistency through a simple application of the continuous mapping theorem. □

Estimator 1 (Miller–Madow)**.** *Let p be a categorical over K categories. We seek to estimate the entropy $\mathrm{H}(p)$. Let $\mathcal{D}$ be our dataset of size N sampled from p. Then, the Miller–Madow estimator of $\mathrm{H}(p)$ is given by*

$$\widehat{\mathrm{H}}_{\text{MM}}(\mathcal{D}) \overset{\text{def}}{=} \widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) + \frac{K-1}{2N} \tag{52}$$

The Miller–Madow estimator is biased, however it is consistent.

Lemma 1. *The the first-order Taylor approximation of $\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D})$ around the distribution p is given by*

$$\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) = \mathrm{H}(\widehat{p}_{\text{MLE}}, p) + R(p, \widehat{p}_{\text{MLE}}) \tag{53}$$

where the remainder R is given by

$$R(p, \widehat{p}_{\text{MLE}}) = -\mathrm{KL}(\widehat{p}_{\text{MLE}} \mid\mid p) \tag{54}$$

Proof. The result follows from direct computation. We start by taking the Taylor expansion of $\mathrm{H}(\widehat{p}_{\text{MLE}})$ around $\mathrm{H}(p)$:

$$\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) = \mathrm{H}(p) + \sum_{k=1}^{K} \frac{\partial}{\partial p(x_k)}\Big[\mathrm{H}(p)\Big]\Big(\widehat{p}_{\text{MLE}}(x_k) - p(x_k)\Big) + \underbrace{R(p, \widehat{p}_{\text{MLE}})}_{\text{remainder}} \tag{55}$$

Our first order term can then be rewritten as follows:

$$\sum_{k=1}^{K} \frac{\partial}{\partial p(x_k)}\Big[\mathrm{H}(p)\Big]\Big(\widehat{p}_{\text{MLE}}(x_k) - p(x_k)\Big) \tag{56}$$

$$= \sum_{k=1}^{K} \frac{\partial}{\partial p(x_k)}\left[\sum_{k'=1}^{K} -p(x_{k'}) \log p(x_{k'})\right]\Big(\widehat{p}_{\text{MLE}}(x_k) - p(x_k)\Big) \tag{57}$$

$$= \sum_{k=1}^{K} \left[\sum_{k'=1}^{K} -\frac{\partial}{\partial p(x_k)} p(x_{k'}) \log p(x_{k'})\right]\Big(\widehat{p}_{\text{MLE}}(x_k) - p(x_k)\Big) \qquad \text{(linearity)} \tag{58}$$

$$= \sum_{k=1}^{K} \left[\sum_{k'=1}^{K} \frac{\partial}{\partial p(x_k)} p(x_{k'}) \log p(x_{k'})\right]\Big(p(x_k) - \widehat{p}_{\text{MLE}}(x_k)\Big) \qquad \text{(sign)} \tag{59}$$

$$= \sum_{k=1}^{K} \Big(1 + \log p(x_k)\Big)\Big(p(x_k) - \widehat{p}_{\text{MLE}}(x_k)\Big) \tag{60}$$

$$= \sum_{k=1}^{K} \Big(p(x_k) - \widehat{p}_{\text{MLE}}(x_k)\Big) + \log p(x_k)\left(p(x_k) - \widehat{p}_{\text{MLE}}(x_k)\right) \tag{61}$$

$$= \sum_{k=1}^{K} \Big(p(x_k) - \widehat{p}_{\text{MLE}}(x_k)\Big) + \sum_{k=1}^{K} \log p(x_k)\left(p(x_k) - \widehat{p}_{\text{MLE}}(x_k)\right) \tag{62}$$

$$= \underbrace{\sum_{k=1}^{K} p(x_k)}_{=1} - \underbrace{\sum_{k=1}^{K} \widehat{p}_{\text{MLE}}(x_k)}_{=1} + \sum_{k=1}^{K} \log p(x_k)\left(p(x_k) - \widehat{p}_{\text{MLE}}(x_k)\right) \qquad \text{(distrib. sum)} \tag{63}$$

$$= \sum_{k=1}^{K} \log p(x_k)\left(p(x_k) - \widehat{p}_{\text{MLE}}(x_k)\right) \qquad \text{(simplify)} \tag{64}$$

$$= \underbrace{\sum_{k=1}^{K} \log p(x_k) p(x_k)}_{-\mathrm{H}(p)} - \underbrace{\sum_{k=1}^{K} \log p(x_k) \widehat{p}_{\text{MLE}}(x_k)}_{\mathrm{H}(p, \widehat{p}_{\text{MLE}})} \qquad \text{(distrib. sum)} \quad (65)$$

$$= \mathrm{H}(p, \widehat{p}_{\text{MLE}}) - \mathrm{H}(p) \quad (66)$$

Plugging this back into our Taylor expansion, we get the following:

$$\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) = \cancel{\mathrm{H}(p)} - \cancel{\mathrm{H}(p)} + \mathrm{H}(p, \widehat{p}_{\text{MLE}}) + R(p, \widehat{p}_{\text{MLE}}) \quad (67)$$

Now, we see that this implies

$$R(p, \widehat{p}_{\text{MLE}}) = \widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) - \mathrm{H}(\widehat{p}_{\text{MLE}}, p) \qquad \text{(algebra)} \quad (68)$$

$$= -\sum_{k=1}^{K} \widehat{p}_{\text{MLE}}(x_k) \log \widehat{p}_{\text{MLE}}(x_k) + \sum_{k=1}^{K} \widehat{p}_{\text{MLE}}(x_k) \log p(x_k) \qquad \text{(defn.)} \quad (69)$$

$$= -\sum_{k=1}^{K} \left(\widehat{p}_{\text{MLE}}(x_k) \log \widehat{p}_{\text{MLE}}(x_k) - \widehat{p}_{\text{MLE}}(x_k) \log p(x_k)\right) \qquad \text{(merge sums)} \quad (70)$$

$$= -\sum_{k=1}^{K} \widehat{p}_{\text{MLE}}(x_k)(\log \widehat{p}_{\text{MLE}}(x_k) - \log p(x_k)) \qquad \text{(factor out } \widehat{p}_{\text{MLE}}(x_k)) \quad (71)$$

$$= -\sum_{k=1}^{K} \widehat{p}_{\text{MLE}}(x_k) \log \frac{\widehat{p}_{\text{MLE}}(x_k)}{p(x_k)} \qquad \text{(log algebra)} \quad (72)$$

$$= -\mathrm{KL}(\widehat{p}_{\text{MLE}} \mid\mid p) \qquad \text{(defn.)} \quad (73)$$

which is the desired result. □

Lemma 2. *Define* $\Delta(x) = p(x) - q(x)$*. The second-order Taylor expansion of* $\mathrm{KL}(p \mid\mid q)$ *around* $\Delta(x)$ *is given by*

$$\mathrm{KL}(p \mid\mid q) = \sum_{x \in \mathcal{X}} \frac{\Delta(x)^2}{2q(x)} + o\left(\Delta(x)^2\right) \quad (74)$$

Proof. Now we compute the series expansion of the KL-divergence. We first make a tricky substitution:

$$\frac{p(x)}{q(x)} = \frac{q(x) + p(x) - q(x)}{q(x)} = 1 + \frac{p(x) - q(x)}{q(x)} = 1 + \frac{\Delta(x)}{q(x)} \quad (75)$$

Now, we proceed with the derivation:

$$\mathrm{KL}(p \mid\mid q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)} \qquad \text{(defn. of KL divergence)} \quad (76)$$

$$= \sum_{x \in \mathcal{X}} (q(x) + \Delta(x)) \log\left(1 + \frac{\Delta(x)}{q(x)}\right) \qquad \text{(Eq. (75))} \quad (77)$$

$$= \sum_{x \in \mathcal{X}} (q(x) + \Delta(x)) \left(\frac{\Delta(x)}{q(x)} - \frac{\Delta(x)^2}{2q(x)^2} + o\left(\Delta(x)^2\right)\right) \qquad \text{(Taylor expansion)} \quad (78)$$

$$= \sum_{x \in \mathcal{X}} \Delta(x) - \frac{\Delta(x)^2}{2q(x)} + \frac{\Delta(x)^2}{q(x)} - \frac{\Delta(x)^3}{2q(x)^2} + o\left(\Delta(x)^2\right) \qquad \text{(distribute)} \quad (79)$$

$$= \sum_{x \in \mathcal{X}} \Delta(x) - \frac{\Delta(x)^2}{2q(x)} + \frac{\Delta(x)^2}{q(x)} + o\left(\Delta(x)^2\right) \qquad \text{(defn. of } o) \quad (80)$$

$$= \sum_{x \in \mathcal{X}} \Delta(x) + \frac{\Delta(x)^2}{2q(x)} + o\left(\Delta(x)^2\right) \qquad \text{(algebra)} \quad (81)$$

$$= \underbrace{\sum_{x\in\mathcal{X}} \Delta(x)}_{=0} + \sum_{x\in\mathcal{X}} \frac{\Delta(x)^2}{2q(x)} + o\left(\Delta(x)^2\right) \qquad \text{(split sums)} \tag{82}$$

$$= \sum_{x\in\mathcal{X}} \frac{\Delta(x)^2}{2q(x)} + o\left(\Delta(x)^2\right) \tag{83}$$

which is the desired result. □

E.3 Jackknife

The jackknife resampling method is used to estimate the bias of an estimator and correct for it, by sampling all subsamples of size $N-1$ from the available sample of size N, computing their average for the statistic being estimated.

Generally, this reduces the order of the bias of an estimator from $O(N^{-1})$ to at most $O(N^{-2})$ (Friedl and Stampfer, 2002).

Estimator 2 (Jackknife)**.** *Let p be a categorical over K categories. We seek to estimate the entropy $\mathrm{H}(p)$. Let $\mathcal{D}$ be our dataset of size N sampled from p. Let $\widehat{\mathrm{H}}^{\setminus n}(\mathcal{D})$ be an estimate of the entropy from a sample with the n^{th} observation held out. Then, the **Jackknife estimator** is given by*

$$\widehat{\mathrm{H}}_{\text{JACK}}(\mathcal{D}) \overset{\text{def}}{=} N\,\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) - \frac{N-1}{N}\sum_{n=1}^{N} \widehat{\mathrm{H}}^{\setminus n}_{\text{MLE}}(\mathcal{D}) \tag{84}$$

This estimator is derived from the jackknife-resampled estimate of the bias of the MLE estimator, multiplied by $N-1$.

$$\widehat{\mathrm{H}}_{\text{JACK}}(\mathcal{D}) - \widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) = (N-1)\left(\widehat{\mathrm{H}}_{\text{MLE}}(\mathcal{D}) - \frac{1}{N}\sum_{n=1}^{N} \widehat{\mathrm{H}}^{\setminus n}_{\text{MLE}}(\mathcal{D})\right) \tag{85}$$

E.4 Horvitz–Thompson

Horvitz and Thompson (HT; 1952) is a common estimator given a finite universe, which is our case as K is finite. We omit a derivation a full here as it is well documented in other places (Vieira, 2017). However, we note that, in contrast to many applications of HT, the application of HT to entropy estimation results in a biased estimator as the function whose mean we seek to estimate is $\log p(x_k)$, which is dependent on the unknown distribution p.

Estimator 3 (Horvitz–Thompson)**.** *Let p be a categorical over K categories. We seek to estimate the entropy $\mathrm{H}(p)$. Let $\mathcal{D}$ be our dataset of size N sampled from p. Then the **Horvitz–Thompson estimator** is defined as*

$$\widehat{\mathrm{H}}_{\text{HT}}(\mathcal{D}) \overset{\text{def}}{=} -\sum_{k=1}^{K} \frac{\widehat{p}_{\text{MLE}}(x_k)\log \widehat{p}_{\text{MLE}}(x_k)}{1-(1-\widehat{p}_{\text{MLE}}(x_k))^N} \tag{86}$$

*where $1-(1-\widehat{p}_{\text{MLE}}(x_k))^N$ is an estimate of the **inclusion probability**, i.e., the probability that x_k appears in a random sample $\mathcal{D}$ of size N.*

We do not know of a simple expression for the bias of the Horvitz–Thompson entropy estimator, but one observation is that $\mathbb{E}_p\left[(1-\widehat{p}_{\text{MLE}}(x_k))^N\right] > \mathbb{E}_p\left[(1-p(x_k))^N\right]$ when $N > 1$ (justified by Jensen's inequality, since x^N, $N>1$ is convex over $[0,1]$); this is an overestimate of the true inclusion probability.

E.5 Chao–Shen

The Chao–Shen estimator builds upon Horvitz–Thompson by noting that that estimator does not correct for underestimation of number of classes K and resulting effect on estimates of $p(x_k)$; i.e. $1-(1-\widehat{p}_{\text{MLE}}(x_k))^N$ is always 0 for a class not included in the sample even if the class is present in the true distribution. We can reweight the sample probabilities to compensate for missing classes using the notion of sample coverage.

Definition 1 (Sample coverage)**.** *We define the* ***sample coverage*** *as*

$$C = \sum_{k=1}^{K} p(x_k) \mathbb{1}\left\{x_k \in \mathcal{D}\right\} \tag{87}$$

Definitionally, $(1 - C)$ *is then the probability of sampling an* x_k not *observed in the sample* $\widetilde{\mathcal{X}}$.

However, exact computation of Eq. (88) is impossible as we do not know the true distribution p. Thus, Chao and Shen (2003) fall back on a well-known estimator of C that uses a technique from Good–Turing (1953) smoothing. Let f_1 be the number of classes with only one observation in the current sample, i.e, the number of singletons, then we can estimate the sample coverage as

$$\widehat{C} \overset{\text{def}}{=} 1 - \frac{f_1}{N} \tag{88}$$

The Chao–Shen estimator, described below, simply re-scales the MLE estimate of probability $\widehat{p}_{\text{MLE}}(x_k)$ in the HT estimator by $\widehat{C}$. This corrects for the observed *under*estimation of p's entropy by HT.

Estimator 4 (Chao–Shen)**.** *Let* p *be a categorical over* K *categories. We seek to estimate the entropy* $\mathrm{H}(p)$. *Let* $\mathcal{D}$ *be our dataset of size* N *sampled from* p. *Let* $\widehat{C}$, *an estimate of sample coverage, be defined as in Eq.* (88). *The* ***Chao–Shen estimator*** *is then defined as*

$$\widehat{\mathrm{H}}_{\text{CS}}(\mathcal{D}) \overset{\text{def}}{=} -\sum_{k=1}^{K} \frac{\widehat{C} \cdot \widehat{p}_{\text{MLE}}(x_k) \log\left(\widehat{C} \cdot \widehat{p}_{\text{MLE}}(x_k)\right)}{1 - (1 - \widehat{C} \cdot \widehat{p}_{\text{MLE}}(x_k))^N} \tag{89}$$

E.6 Wolpert–Wolf

Fact 1 (Derivative of an exponent)**.**

$$\frac{\mathrm{d}}{\mathrm{d}a} x^a = x^a \log x \tag{90}$$

Fact 2 (Normalizer of a Dirichlet)**.** *The normalizer of a Dirichlet distribution is*

$$\int \delta\left(\sum_{k=1}^{K} x_k - 1\right) \prod_{k=1}^{K} x^{\alpha_k}\, \mathrm{d}\boldsymbol{x} = \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma\left(\sum_{k=1}^{K} \alpha_k\right)} \tag{91}$$

A relatively easy proof of this fact makes use of a Laplace transform.

Estimator 5 (Wolpert–Wolf)**.** *Let* p *be a categorical over* K *categories. We seek to estimate the entropy* $\mathrm{H}(p)$. *Let* $\mathcal{D}$ *be our dataset of size* N *sampled from* p. *Then, the* ***Wolpert–Wolf estimator*** *is given by*

$$\widehat{\mathrm{H}}_{\text{WW}}(\mathcal{D} \mid \boldsymbol{\alpha}) \overset{\text{def}}{=} \psi\left(\widetilde{A} + 1\right) - \sum_{k=1}^{K} \frac{\widetilde{\alpha}_k}{\widetilde{A}} \psi(\widetilde{\alpha}_k + 1) \tag{92}$$

where $c(x_k) \overset{\text{def}}{=} \sum_{n=1}^{N} \mathbb{1}\{\widetilde{x}_n = x_k\}$, *and we additionally define* $\widetilde{\alpha}_k \overset{\text{def}}{=} c(x_k) + \alpha_k$ *and* $\widetilde{A} \overset{\text{def}}{=} \sum_{k=1}^{K} \widetilde{\alpha}_k$.

Proposition 3 (Wolpert–Wolf)**.** *The expectation of entropy under a Dirichlet posterior* $\mathrm{Dirichlet}(\boldsymbol{\alpha})$ *where parameter* $\boldsymbol{\alpha}$ *is given by*

$$\mathbb{E}\left[\mathrm{H}(p) \mid \boldsymbol{\alpha}\right] \overset{\text{def}}{=} \int \mathrm{H}(p)\, \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) \frac{\Gamma(A)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} p(x_k)^{\alpha_k - 1} \mathrm{d}p \tag{93}$$

$$= \psi(A + 1) - \sum_{k=1}^{K} \frac{\alpha_k}{A} \psi(\alpha_k + 1) \tag{94}$$

where $A \overset{\text{def}}{=} \sum_{k=1}^{K} \alpha_k$.

Proof. Let $\text{Dirichlet}(\alpha_1, \ldots, \alpha_K)$ be a Dirichlet posterior. The result follows by a series of manipulations:

$$\mathbb{E}\left[\mathrm{H}(p) \mid \boldsymbol{\alpha}\right] = \int \mathrm{H}(p)\, \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) \frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \prod_{k=1}^{K} p(x_k)^{\alpha_k - 1} \mathrm{d}p \qquad \text{(defn.)} \tag{95}$$

$$= \frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \int \mathrm{H}(p)\, \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) \prod_{k=1}^{K} p(x_k)^{\alpha_k - 1} \mathrm{d}p \tag{96}$$

$$= \frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \int \left(-\sum_{k=1}^{K} p(x_k) \log p(x_k)\right) \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) \prod_{k=1}^{K} p_k^{\alpha_k - 1} \mathrm{d}p \qquad \text{(defn. H)} \tag{97}$$

$$= -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \int p(x_k) \log p(x_k) \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) \prod_{k=1}^{K} p(x_k)^{\alpha_k - 1} \mathrm{d}p \qquad \text{(linear.)} \tag{98}$$

$$= -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \int p(x_k)^{\alpha_k} \log p(x_k) \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) \prod_{\substack{j=1,\\ j\neq k}}^{K} p(x_j)^{\alpha_j - 1} \mathrm{d}p \qquad \text{(algebra)} \tag{99}$$

$$= -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \int \frac{\mathrm{d}}{\mathrm{d}\alpha_k} p(x_k)^{\alpha_k} \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) \prod_{\substack{j=1,\\ j\neq k}}^{K} p(x_j)^{\alpha_j - 1} \mathrm{d}p \qquad \text{(fact \#1)} \tag{100}$$

$$= -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \int \frac{\mathrm{d}}{\mathrm{d}\alpha_k} \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) p(x_k)^{\alpha_k} \prod_{\substack{j=1,\\ j\neq k}}^{K} p(x_j)^{\alpha_j - 1} \mathrm{d}p \qquad \text{(algebra)} \tag{101}$$

$$= -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \frac{\mathrm{d}}{\mathrm{d}\alpha_k} \int \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) p(x_k)^{\alpha_k} \prod_{\substack{j=1,\\ j\neq k}}^{K} p(x_j)^{\alpha_j - 1} \mathrm{d}p \tag{102}$$

$$= -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \frac{\mathrm{d}}{\mathrm{d}\alpha_k} \frac{\Gamma(\alpha_k + 1) \prod_{\substack{j=1,\\ j\neq k}}^{K} \Gamma(\alpha_j)}{\Gamma\left(\sum_{j=1}^{K} \alpha_j + 1\right)} \qquad \text{(fact \#2)} \tag{103}$$

$$= -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \prod_{\substack{j=1,\\ j\neq k}}^{K} \Gamma(\alpha_j) \frac{\mathrm{d}}{\mathrm{d}\alpha_k} \frac{\Gamma(\alpha_k + 1)}{\Gamma\left(\sum_{j=1}^{K} \alpha_j + 1\right)} \tag{104}$$

$$\begin{aligned} = -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \prod_{\substack{j=1,\\ j\neq k}}^{K} \Gamma(\alpha_j) & \frac{\psi(\alpha_k + 1)\Gamma(\alpha_k + 1)\Gamma\left(\sum_{j=1}^{K} \alpha_j + 1\right)}{\Gamma\left(\sum_{j=1}^{K} \alpha_j + 1\right)^2} \\ & - \frac{\psi(\sum_{j=1}^{K} \alpha_j + 1)\Gamma(\alpha_k + 1)\Gamma(\sum_{j=1}^{K} \alpha_k + 1)}{\Gamma\left(\sum_{j=1}^{K} \alpha_j + 1\right)^2} \end{aligned} \qquad \text{(derivative)} \tag{105}$$

$$= -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \prod_{\substack{j=1,\\ j\neq k}}^{K} \Gamma(\alpha_j) \frac{\begin{array}{c}\psi(\alpha_k + 1)\Gamma(\alpha_k + 1) \\ - \psi(\sum_{j=1}^{K} \alpha_j + 1)\Gamma(\alpha_k + 1)\end{array}}{\Gamma\left(\sum_{j=1}^{K} \alpha_j + 1\right)} \qquad \text{(simplify)} \tag{106}$$

$$= -\frac{\Gamma\left(A\right)}{\prod_{k=1} \Gamma(\alpha_k)} \sum_{k=1}^{K} \prod_{\substack{j=1,\\ j\neq k}}^{K} \Gamma(\alpha_j) \frac{\begin{array}{c}\psi(\alpha_k + 1)\Gamma(\alpha_k)\alpha_k \\ - \psi(\sum_{j=1}^{K} \alpha_j + 1)\Gamma(\alpha_k)\alpha_k\end{array}}{\Gamma\left(\sum_{j=1}^{K} \alpha_j\right) A} \qquad \text{(defn. }\Gamma\text{)} \tag{107}$$

$$= -\frac{\Gamma(A)}{\prod_{k=1}\Gamma(\alpha_k)} \frac{\prod_{k=1}^{K}\Gamma(\alpha_k)}{\Gamma(A)} \sum_{k=1}^{K}\left(\frac{\alpha_k}{A}\psi(\alpha_k+1) - \frac{\alpha_k}{A}\psi\left(\sum_{k=1}^{K}\alpha_k+1\right)\right) \quad \text{(distrib.)} \tag{108}$$

$$= -\sum_{k=1}^{K}\left(\frac{\alpha_k}{A}\psi(\alpha_k+1) - \frac{\alpha_k}{A}\psi\left(\sum_{k=1}^{K}\alpha_k+1\right)\right) \quad \text{(cancel)} \tag{109}$$

$$= -\sum_{k=1}^{K}\left(\frac{\alpha_k}{A}\psi(\alpha_k+1) - \frac{\alpha_k}{A}\psi(A+1)\right) \quad \text{(defn. } A\text{)} \tag{110}$$

$$= -\sum_{k=1}^{K}\frac{\alpha_k}{A}\psi(\alpha_k+1) + \sum_{k=1}^{K}\frac{\alpha_k}{A}\psi(A+1) \quad \text{(distrib.)} \tag{111}$$

$$= -\sum_{k=1}^{K}\frac{\alpha_k}{A}\psi(\alpha_k+1) + \psi(A+1) \quad (\textstyle\sum a_k = A) \tag{112}$$

$$= \psi(A+1) - \sum_{k=1}^{K}\frac{\alpha_k}{A}\psi(\alpha_k+1) \quad \text{(rearr.)} \tag{113}$$

which proves the result. □

E.7 Nemenman–Shafee–Bialek

Estimator 6 (Nemenman–Shafee–Bialek)**.** *Let p be a categorical over K categories. We seek to estimate the entropy $\mathrm{H}(p)$. Let $\mathcal{D}$ be our dataset of size N sampled from p. Define the NSB density as*

$$p_{\text{NSB}}(\alpha) \stackrel{\text{def}}{=} \frac{K\psi_1(K\alpha+1) - \psi_1(\alpha+1)}{\log K} \tag{114}$$

where ψ_1 is the trigramma function. Then, the ***NSB estimator*** *is given by*

$$\widehat{\mathrm{H}}_{\text{NSB}}(\mathcal{D}) \stackrel{\text{def}}{=} \int_0^\infty \widehat{\mathrm{H}}_{\text{WW}}(\mathcal{D} \mid \alpha \cdot \mathbf{1})\, p_{\text{NSB}}(\alpha)\, \mathrm{d}\alpha \tag{115}$$

The integral in Eq. (115) is typically computed by numerical integration.

To derive the Nemenman–Shafee–Bialek (NSB) estimator, we start with the idea that we would like a prior over distributions such that the distribution over expected entropy is uniform. In other words, we are looking for a p_{NSB} such that for $\alpha \sim p_{\text{NSB}}$, the values of $\mathbb{E}_p[\mathrm{H}(p) \mid \alpha]$ are uniformly distributed over $[0, \log K]$. This is a good idea since, a-priori, we do not know entropy of p and, in the absence of any insight, we should assume the entropy could be anywhere in the range $[0, \log K]$. We make the above intuition formal with the following proposition.

Proposition 4. *Let p_{NSB} be the NSB density given in Eq. (114). Then the following conditional expectation*

$$\mathbb{E}_p[\mathrm{H}(p) \mid \alpha] \stackrel{\text{def}}{=} \int \mathrm{H}(p)\, \delta\left(\sum_{k=1}^{K} p(x_k) - 1\right) \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \prod_{k=1}^{K} p(x_k)^{\alpha-1}\, \mathrm{d}p \tag{116}$$

$$= \psi(K\alpha+1) - \psi(\alpha+1) \quad \textit{(Proposition 3)} \tag{117}$$

is uniformly distributed over $[0, \log K]$ when $\alpha \sim p_{\text{NSB}}(\cdot)$, defined in Eq. (114).

Proof. First, we note that $\mathbb{E}_p[\mathrm{H}(p) \mid \alpha]$ is a continuous, increasing function in α. We will not prove this formally, but it should make intuitive sense: α is a smoothing parameter and the more the distribution is smoothed, the more entropic it should be. From basic analysis, we know that a strictly continuous, increasing function has an inverse. The above means that we can view $\mathbb{E}_p[\mathrm{H}(p) \mid \alpha]$ as a bijection from $\mathbb{R}_{\geq 0}$ to the interval $[0, \log K]$. Our goal is to reparameterize the Uniform distribution in terms of α. To that end, we define the function $g^{-1}(\alpha) \overset{\text{def}}{=} \mathbb{E}_p\left[\mathrm{H}(p) \mid \alpha\right] : \mathbb{R}_{\geq 0} \to [0, \log K]$ and perform a change-of-variables transform on Eq. (118) using g^{-1}. We start with the continuous uniform over $[0, \log K]$, which is show below

$$p(H) \overset{\text{def}}{=} \underbrace{\frac{1}{\log K} \mathbb{1}\left\{H \in [0, \log K]\right\}}_{\text{uniform over } [0, \log K]} \qquad \text{(defn. of uniform dist)} \qquad (118)$$

Note H is a random variable and unrelated to the functional $\mathrm{H}(\cdot)$; the choice of letter intentionally reminds one that the variable represents the expected entropy of under a random distribution. Now we apply the change-of-variables formula at $H = g^{-1}(\alpha)$ and manipulate:

$$\begin{aligned}
p(H) &= p(g^{-1}(\alpha)) \left|\frac{\mathrm{d}g^{-1}}{\mathrm{d}\alpha}(\alpha)\right| && \text{(change of variable)} && (119)\\
&= \frac{1}{\log K} \mathbb{1}\left\{g^{-1}(\alpha) \in [0, \log K]\right\} \left|\frac{\mathrm{d}g^{-1}}{\mathrm{d}\alpha}(\alpha)\right| && \text{(definition of } p) && (120)\\
&= \frac{1}{\log K} \left|\frac{\mathrm{d}g^{-1}}{\mathrm{d}\alpha}(\alpha)\right| && \text{(redundant indicator)} && (121)\\
&= \frac{1}{\log K} \frac{\mathrm{d}g^{-1}}{\mathrm{d}\alpha}(\alpha) && \text{(derivative is positive)} && (122)\\
&= \frac{K\psi_1\left(K\alpha + 1\right) - \psi_1(\alpha + 1)}{\log K} && \text{(Lemma 3)} && (123)\\
&\overset{\text{def}}{=} p_{\text{NSB}}(\alpha) && \text{(definition)} && (124)
\end{aligned}$$

By construction, the prior $p_{\text{NSB}}(\alpha)$ has the property that the expected entropy $\mathbb{E}_p\left[\mathrm{H}(p) \mid \alpha\right]$ where $\alpha \sim p_{\text{NSB}}(\cdot)$ is uniformly distributed over $[0, \log K]$, which we can see by reversing the above derivation. This proves the result. □

Nemenman et al. (2002) interpreted Proposition 4 in the following manner: As the variance of $\mathbb{E}_p\left[\mathrm{H}(p) \mid \alpha\right]$, which is treated as a random variable since α is random, approaches 0, then the the NSB estimator implies a uniform prior over the entropy.

Lemma 3 (NSB Derivative)**.**

$$\frac{\mathrm{d}}{\mathrm{d}\alpha}\left[\psi(K\alpha + 1) - \psi(\alpha + 1)\right] = K\psi_1(K\alpha + 1) - \psi_1(\alpha + 1) \qquad (125)$$

Proof. The proof follows by a straightforward computation:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\alpha}\left[\psi(K\alpha + 1) - \psi(\alpha + 1)\right] &= \frac{\mathrm{d}}{\mathrm{d}\alpha}\left[\psi(K\alpha + 1)\right] - \frac{\mathrm{d}}{\mathrm{d}\alpha}\left[\psi(\alpha + 1)\right] && \text{(linearity)} && (126)\\
&= K\psi_1(K\alpha + 1) - \psi_1(\alpha + 1) && \text{(definition)} && (127)
\end{aligned}$$

where $\psi_1(x) \overset{\text{def}}{=} \frac{\mathrm{d}}{\mathrm{d}x}\psi(x)$. □

Morphological Reinflection with Multiple Arguments: An Extended Annotation schema and a Georgian Case Study

David Guriel, Omer Goldman, Reut Tsarfaty
Bar-Ilan University
{davidgu1312,omer.goldman}@gmail.com,reut.tsarfaty@biu.ac.il

Abstract

In recent years, a flurry of morphological datasets had emerged, most notably UniMorph, a multi-lingual repository of inflection tables. However, the flat structure of the current morphological annotation schema makes the treatment of some languages quirky, if not impossible, specifically in cases of polypersonal agreement, where verbs agree with multiple arguments using true affixes. In this paper we propose to address this phenomenon, by expanding the UniMorph annotation schema to hierarchical feature structure that naturally accommodates complex argument marking. We apply this extended schema to one such language, Georgian, and provide a human-verified, accurate and balanced morphological dataset for Georgian verbs. The dataset has 4 times more tables and 6 times more verb forms compared to the existing UniMorph dataset, covering all possible variants of argument marking, demonstrating the adequacy of our proposed scheme. Experiments with a standard reinflection model show that generalization is easy when the data is split at the form level, but extremely hard when splitting along lemma lines. Expanding the other languages in UniMorph to this schema is expected to improve both the coverage, consistency and interpretability of this benchmark.

1 Introduction

In recent years, morphological (re)inflection tasks have gained a lot of attention in NLP.[1] Subsequently, several multi-lingual morphological datasets have emerged to allow for the supervised training of morphological models, most notably UniMorph (McCarthy et al., 2020), that organizes words into inflectional tables, annotating each inflected word-form with its respective feature-set.

While western languages are widely represented in UniMorph, many *morphologically rich languages* (Tsarfaty et al., 2010, 2020) exhibit rich and diverse inflection patterns that make them less compatible with the flat feature-sets in the UniMorph schema. Concretely, in some cases it is completely impossible to annotate parts of the inflectional paradigm with a flat bundle, as is the case with *case stacking*, and in other cases, such as *polypersonal agreement*, the annotation solutions provided are unnatural, non-transparent, and are barely used in practice. As a result, languages exhibiting such phenomena are under-represented in UniMorph, and when they are, the inflection tables for these languages are often incomplete.

In this paper we propose a general solution for annotating such structures, thus extending the UniMorph annotation schema to fully cover a wider range of morphologically-complex argument-marking phenomena. Following Anderson (1992), we propose a so-called *layered* annotation of features, where the inflectional features take the form of a *hierarchical* structure, in the spirit of formal linguistic frameworks as that of Johnson (1988); Pollard and Sag (1994); Shieber (2003); Bresnan et al. (2015). We organize the features of multiple arguments in a hierarchical structure, rather than the current flat structure that accommodates only subject concords. This schema shift allows for an adequate annotation of *polypersonal agreement* and of *possessed nominals*, where a word has multiple number and gender features, as well as forms with *case stacking*, where a word has multiple cases.

We apply the suggested solution to Georgian, an agglutinative language with a convoluted verbal system, that indicates both subjects and objects with true affixes (rather than clitics that are omittable from the inflection tables). We create a new human-verified dataset for Georgian, that covers most of the grammatical phenomena in Georgian verbs, and includes 118 lemmas, adding up to about $21k$ verb forms, compared with the 47 lemmas and $3.3k$ verb forms, some of which are erroneous, currently available in the Georgian UniMorph.

[1] Cf. the series of SIGMORPHON shared tasks: https://sigmorphon.github.io/sharedtasks/

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 196 - 202
May 22-27, 2022

გაგიშვებთ (*gagišvebt*)					
გა-	გ-	ი-	შვ	-ებ	-თ
ga-	g-	i-	šv	-eb	t
FUT	O2SG	TRANS	LET GO	THEME	S1PL
We will let you(sg.) go					

Table 1: A typical Georgian verb. Note the 2 argument markers, one object (tagged with O) and one subject (S).

We use the new dataset to train a standard morphological reinflection model (Silfverberg and Hulden, 2018) and show that training on the Georgian inflections currently available in UniMorph is not sufficient for generalizing to the more inclusive set of inflections that are allowed by the new scheme. We conclude that our annotation approach provides a more complete representation of linguistic behaviors, and that our proposed Georgian dataset provides a much better depiction of the morphological phenomena that exist in the data and the computational challenge reflected therein.

We therefore call to apply layered annotation to all currently existing morphological data in UniMorph, to more consistently and transparently capture the linguistic reality and morphological complexity reflected in the worlds languages.

2 The Problem: Multiple Arguments

Models of morphological reinfection are trained to generate forms within a lemma L, given another form and the features of source$_i$ and target$_j$ forms:

$$(\langle feat_i^L, form_i^L\rangle, \langle feat_j^L, \_\_\rangle) \mapsto form_j^L$$

For example, for the Russian lemma ЛЕТЕТЬ: reinflecting from (PRS;1;SG,лечу) to (IMP;2;SG,лети) will be represented as:

$$(\langle \text{PRS;1;SG}, \text{лечу}\rangle, \langle \text{IMP;2;SG}, \_\_\rangle) \mapsto \text{лети}$$

Standardly, the data for training morphological models (e.g., Wu et al., 2020; Makarov and Clematide, 2018) is taken from UniMorph (McCarthy et al., 2020), a multilingual morphological dataset in which words are grouped by lemma into inflection tables, each word is tagged with an unordered set of morphological features. The features list is shared across languages. The inflection tables are meant to be exhaustive, i.e., covering all possible forms of a lemma, regardless of usability.

Although the features were designed to apply cross-lingually, some blind-spots exist. Most relevant to our work is the assumption that every feature set includes at most one pronominal feature bundle (i.e., person-gender-number). However, this assumption does not apply to verbs with object concords, as exhibited in Georgian (see Table 1), Inuit and many Bantu languages *inter alia*, nor does it apply to possessed nouns that mark the features of both the possessor and the possessee. Examples (1a)–(1d) illustrate this:

(1) a. Georgian: *gagišvebt* ‘We will let you go’
(SUBJ-1PL, OBJ-2SG)
b. Turkish: *kedisisin* ‘you are his cat’
(NOUN-SG, SUBJ-2SG, POSS-3SG)
c. Swahili: *ninakupenda* ‘I love you’
(SUBJ-1SG, OBJ-2SG)
d. Hebrew: *emdata* ‘her position’
(NOUN-SG, POSS-3SG-FEM)

The solution proposed in UniMorph to annotating these phenomena is via concatenating several properties into a single string, lacking any internal structure; e.g., ARGAC2S indicates a form with a 2nd person singular accusative argument (Sylak-Glassman, 2016). However, there are at least two shortcomings to this solution. First, it is not sufficiently transparent. ARGAC2S is an opaque string, that does not decompose into the known features licensed by the UniMorph features list (i.e., ACC, 2, SG). Secondly, and possibly due to this lack of transparency, this annotation hack is hardly ever used in practice. Hence, from all examples in (1), only the Hebrew form is included in UniMorph, and tagged as N;SG;FEM;PSS3S with multiple possessor features merged into the flat string PSS3S.

The crux of the matter is that in the current annotation schema, complex features assigned to additional arguments are treated as a single non-decomposable feature, that lack any internal structure, unlike the features of the main (so-called ‘internal’) argument, that are individually spelled out. We argue that the lack of transparency and usability are due to the misrepresentation of the inherently *hierarchical* and *compositional* structure of the features in such forms. We suggest to explicitly annotate these forms with features that are all explicitly composed of the same primitive features.

All in all, the lack of a sufficiently expressive annotation standard leads to a data distribution that is skewed, unrealistically simple, and, when language-specific annotation solutions are painfully needed, they suffer from inconsistencies and ad-hoc decisions. For these reasons, we set out to extend the UniMorph annotation schema to accommodate all such cases and to enable a proper coverage of languages, such as Georgian and many others.

3 The Proposed Schema

We propose to extend the UniMorph annotation schema to cover multiple pronominal feature-bundles in the same word-form, via a *layering* approach, originally proposed for morphological systems by Anderson (1992). Anderson suggests to arrange the *morphosyntactic representation* (MSR) of words in a hierarchy (dubbed *layers*) of features, in the sense that every element of the unordered set of features can be composed of another unordered set of features. That is, a general feature annotation looks as in (2a). A specific transitive verb annotation could be as depicted in (2b):

(2) a. $[f_1, f_2, ..., [F_i : f_{i_1}, f_{i_2}, ...[F_j : f_{j_1}..]]]$
b. $[V, Tense,$
$[nom : Per, Num, Gen],$
$[acc : Per, Num, Gen]]$

This hierarchical feature structure is reminiscent of *unification grammars* or *attribute-value grammars* (Shieber, 2003; Johnson, 1988) that are extensively used in syntactic theories such as GPSG, HPSG, and resemble the f-structures in LFG (Gazdar et al., 1989; Pollard and Sag, 1994; Bresnan et al., 2015).

Here we employ these structures to organize the features of morphologically-marked arguments hierarchically, so an argument is characterized by a feature composite of all features pertaining to that argument. That is, each argument's feature-bundle os specifically marked with the argument it belongs to, and is decomposed into the primitive features licensed by the UniMorph scheme. It also homogeneously annotates the different kinds of arguments, in contrast with the current schema where the subject features are assigned to the verb directly. Thus, the English form *thinks* previously annotated as V;PRS;3;SG will be annotated as V;PRS;NOM(3;SG). In languages that mark multiple arguments, different kinds of arguments can be marked with their feature-bundles without conflicts. The proposed schema thus facilitates the annotation of the poorly-treated or untreated phenomena as illustrated in (1). These are, respectively:

(3) a. Georgian: *gagišvebt* 'We will let you go'
V;FUT;NOM(1;PL);ACC(2;SG)
b. Turkish: *kedisisin* 'you are his cat'
N;SG;NOM(2;SG);POSS(3;SG)
c. Swahili: *ninakupenda* 'I love you'
V;PRS;NOM(1;SG);ACC(2;SG)
d. Hebrew: *emdata* 'her position'
N;SG;POSS(3;SG;FEM)

Table 2 compares the annotation of these examples in the current UniMorph schema compared with our proposed annotation schema.[2] The hierarchical structures, beyond being more transparent, opens the door further for future study on compositional generalization in morphology.

The resemblance of our proposed schema to ideas in other fields of theoretical linguistics, most prominently to the *f-structure* in LFG (Bresnan et al., 2015) and to the nested *Attribute-Value matrices* in HPSG (Pollard and Sag, 1994), points to a natural interface with further syntactic and semantic annotations downstream.

4 A Case Study from Georgian

Linguistic Background Georgian is an agglutinative language with a verbal system that makes a vast use of affixes to convey a wide array of meanings, both inflectional and derivational (see Table 1). The Georgian verbal paradigm is divided into 5 classes known as: transitive, intransitive, medial, indirect and stative (Hewitt, 1995). The verbs are inflected to reflect 12 Tense-Aspect-Mood (TAM) combinations (traditionally known as *screeves*) sorted into 4 series: present and future, aorist, perfective, and the imperative. Each series has its own morpho-syntactic characteristics, most notably split-ergativity is manifested in the aorist.

The characteristic most essential to this work is that Georgian verbs always agree on person and number with the direct and indirect objects, on top of the subject-verb agreement. The Georgian data in UniMorph follows the convention of including objects *only* in third person singular — thus failing to provide a comprehensive coverage of the word-forms that can be attested in the language.

Additional issues with the current morphological data in UniMorph for Georgian verbs are: sparsity, as it includes only 47 inflection tables; lack of diversity, as all table are from the transitive class; and lack of accuracy, as the data was produced automatically without verification by native speakers.

Data Annotation A key contribution of this work is the creation of a new dataset for Georgian that follows the layered annotation schema and addresses the other shortcomings just described. We selected a list of 118 verb lemmata from all differ-

[2] Although not explicitly shown here, annotation of case stacking is also possible with our approach, while non-hierarchical annotations do not account for such cases. For example, Korean 교사에게이 can be tagged as N;SG;NOM(DAT).

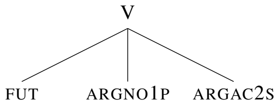
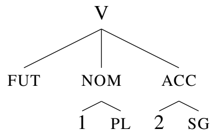
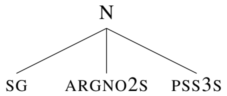
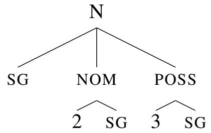
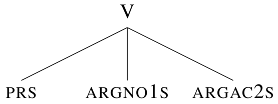
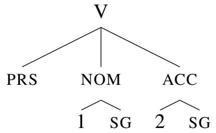
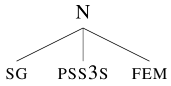
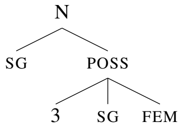
	Flat structure	Hierarchical Structure
Georgian: *gagišvebt* Trans: 'We will let you go' Args: SUBJ-1PL, OBJ-2SG		
Turkish: *kedisisin* Trans: 'you are his cat' Args: NOUN-SG, SUBJ-2SG, POSS-3SG		
Swahili: *ninakupenda* Trans: 'I love you' Args: SUBJ-1SG, OBJ-2SG		
Hebrew: *emdata* Trans: 'her position' Args: NOUN-SG, POSS-3SG-FEM		

Table 2: Examples for word-forms with multiple argument agreements. On the left we present the flat structure currently employed in UniMorph. All examples save Hebrew are not included in the UniMorph inflection tables, presumably due to their lack of transparency. On the right we present our proposed hierarchical structure, which is more transparent, and also ammenable for compositional generalization.

ent classes.[3] Every verb was manually annotated with its stem, its thematic affix and principal parts, to automatically generate the full inflection tables.

This automatic generation of Georgian verbs is prone to some errors, for instance, in accounting for idiosyncratic phonologically-conditioned stem changes. Hence, we ran our data through 3 native Georgian speakers to assert its correctness, or fix when needed. In cases where speakers were unsure we used a Georgian morphological analyzer (Doborjginidze and Lobzhanidze, 2012) for consultation. In cases of disagreement, we used a majority vote among the speakers. On average, at least one speaker was uncertain in about 5% of the forms, but a disagreement that necessitated a majority vote occurred only on about 0.7% of the cases.

Table 3 summarizes the statistics over our annotated data. In total, we produced 21,054 verb forms, of 118 lemmata. The data is quite evenly balanced across the classes, with more verbs drawn from the more frequent transitive class. For comparison, the current UniMorph data has fewer lemmas, 3,300 forms, and includes only verbs that are transitive.[4]

	Trans.	Intrans.	Med.	Indi.	Stat.
#Infl. Tables	40	21	29	16	12
#Verb Forms	12506	2560	3132	2626	230

Table 3: Distribution of the Georgian verbs over classes.

5 Experiments

To assess the usability of our dataset, we trained a standard reinflection model, the character-level LSTM of Silfverberg and Hulden (2018), on our data.[5] We sampled from our data 2 datasets for training morphological reinflection models, containing train, validation and test sets in sizes $8k$, $1k$ and $1k$ examples, respectively. Following Goldman et al. (2021), one dataset employed an easier form-split, i.e., no forms appear in both train and test,[6] and the other with the more challenging lemma-split, where lemmas from train, dev and test are disjoint. To assess the generalization capacity we varied the sources of both the train and test sets.[7] We report 2 evaluation metrics: *accuracy* over exact matches, and *average edit distance* from gold.

[3] We based the list of verbs on those whose inflection tables appear on Hewitt (1995) and added some commonly-used verbs suggested by native speakers.

[4] All our data is publicly available at `https://github.com/Onlp/GeorgianMorphology`.

[5] For hyper-parameters tuning see Appendix C.

[6] This is the splitting method used in SIGMORPHON's shared tasks on reinflection (e.g., Cotterell et al., 2018).

[7] We harmonized the train and test features vocabulary, so that the old data bears the new scheme. So the only difference between Original and New is in which forms are included.

Train Set	Test set	Form Split		Lemma Split	
		Acc	**Avg ED**	**Acc**	**Avg ED**
New	New	94.9%	0.15	1.3%	4.66
New	Original	84.7%	0.3	0.3%	4.39
Original	New	35.2%	1.36	0.0%	6.22
Original	Original	99.3%	0.01	0.0%	6.13

Table 4: Accuracy (**Acc**, higher is better) and Average Edit Distance (**Avg ED**, lower is better) for morphological reinflection on different train-test combinations.

Results and Analysis Table 4 presents the model's performance for all train-test combinations. It shows that the model's performance on the new data (top line combination) is largely on par comparing to its performance over training and testing on UniMorph's original data (bottom combination). However, the model generalizes poorly from the original partial data to the forms in our test set which reflect the entire Georgian inflectional system. Generalization from our data to UniMorph's set is a lot better. The results also show that the splitting method is crucial for success of the model, as it inflects easily to unseen forms, but much harder when inflecting forms in a previously unseen lemma.[8] These results corroborate the results of Goldman et al. (2021) regarding the difficulty of lemma-split data. Although the accuracy over the lemma split data is negligible, the average edit distance in that case points again to the conclusion that generalization from UniMorph to our data is harder that the other way around.

Error Analysis To provide insights into the challenge of reinflecting morphologically complex forms, we manually sampled the erroneous output of the model trained and tested over our lemma-split data, to draw insights on the points of failure. In many cases the model succeeded in copying and modifying the verb stem, but failed to output the other morphemes correctly. Sometimes the errors were due to inflection to an incorrect TAM combination of the same lexeme, and sometimes the inflection was done to the correct TAM but to a different derivationally-related lemma (e.g. change of voice in addition to the change of TAM). We conclude that the fact that our datasets include lemmas from diverse classes that may have derivational relations makes the inflection task significantly harder. Interestingly, the model managed to predict the correct subject and object affixes most of the time.

6 Conclusion

This paper proposes a transition of the UniMorph annotation standard to a layered hierarchical annotation of features. This revised schema caters for complex marking phenomena including multiple pronominal agreement. We apply it to Georgian, and construct a corresponding new dataset that is large, balanced, complete with respect to grammatical phenomena in the Georgian verb system and verified by native-speakers. Our experiments with a standard reinflection model on the old and new Georgian datasets shows that the old UniMorph dataset does not generalize well to the new test-set, due to its partial coverage. This work is intended to encourage the community to extend the annotation of different languages to include phenomena such as polypersonal agreement and others that can be dealt with using a hierarchical annotation, ultimately leading to more complete and consistent benchmarks for studying non-trivial and less-explored areas of computational morphology.

Acknowledgements

The first author would like to thank the native Georgian speakers: Simon Guriel, Silvia Guriel-Agiashvili and Nona Atanelov for their invaluable help in the data annotation process. This research was funded by the European Research Council under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 677352) and by a research grant from the Ministry of Science and Technology (MOST) of the Israeli Government, for which we are grateful.

References

Stephen R Anderson. 1992. *A-morphous morphology*. 62. Cambridge University Press.

Joan Bresnan, Ash Asudeh, Ida Toivonen, and Stephen Wechsler. 2015. *Lexical-functional syntax*. John Wiley & Sons.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

[8] For learning curves on the splits see Appendix A.

Nino Doborjginidze and Irina Lobzhanidze. 2012. Corpus of the georgian language - morphological analyzer. http://corpora.iliauni.edu.ge/?q=search-words.

Nino Doborjginidze and Irina Lobzhanidze. 2016. Corpus of the Georgian Language. In *Proceedings of the XVII EURALEX International Congress*, pages 328–335, Tbilisi, Georgia. Ivane Javakhishvili Tbilisi University Press.

Gerald Gazdar, Ewan Klein, Geoffrey Pullum, and Ivan Sag. 1989. Generalized phrase structure grammar. *Philosophical Review*, 98(4):556–566.

Omer Goldman, David Guriel, and Reut Tsarfaty. 2021. (un)solving morphological inflection: Lemma overlap artificially inflates models' performance.

B. G. Hewitt. 1995. *Georgian a structural reference grammar / B.G. Hewitt.* London Oriental and African language library, v. 2. John Benjamins Pub., Amsterdam.

Mark Johnson. 1988. *Attribute-value logic and the theory of grammar*. Center for the Study of Language and Information.

Peter Makarov and Simon Clematide. 2018. Neural transition-based string transduction for limited-resource setting in morphology. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 83–93, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Arya D. McCarthy, Christo Kirov, Matteo Grella, Amrit Nidhi, Patrick Xia, Kyle Gorman, Ekaterina Vylomova, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, Timofey Arkhangelskiy, Nataly Krizhanovsky, Andrew Krizhanovsky, Elena Klyachko, Alexey Sorokin, John Mansfield, Valts Ernštreits, Yuval Pinter, Cassandra L. Jacobs, Ryan Cotterell, Mans Hulden, and David Yarowsky. 2020. UniMorph 3.0: Universal Morphology. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3922–3931, Marseille, France. European Language Resources Association.

Carl Pollard and Ivan A Sag. 1994. *Head-driven phrase structure grammar*. University of Chicago Press.

Stuart M Shieber. 2003. *An introduction to unification-based approaches to grammar*. Microtome Publishing.

Miikka Silfverberg and Mans Hulden. 2018. An encoder-decoder approach to the paradigm cell filling problem. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2883–2889, Brussels, Belgium. Association for Computational Linguistics.

John Sylak-Glassman. 2016. The composition and use of the universal morphological feature schema (unimorph schema). *Johns Hopkins University*.

Reut Tsarfaty, Dan Bareket, Stav Klein, and Amit Seker. 2020. From SPMRL to NMRL: What did we learn (and unlearn) in a decade of parsing morphologically-rich languages (MRLs)? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7396–7408, Online. Association for Computational Linguistics.

Reut Tsarfaty, Djamé Seddah, Yoav Goldberg, Sandra Kuebler, Yannick Versley, Marie Candito, Jennifer Foster, Ines Rehbein, and Lamia Tounsi. 2010. Statistical parsing of morphologically rich languages (SPMRL) what, how and whither. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 1–12, Los Angeles, CA, USA. Association for Computational Linguistics.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2020. Applying the transformer to character-level transduction.

A Learning Curves

Fig. 1 exemplifies the sufficiency of our dataset for training an inflection model on form-split data as doubling the data amount from 4,000 to 8,000 yields relatively minor improvement. It also shows that for the lemma-split data, the model completely fails. It starts improve marginally with more than 2,000 examples, although its performance remains far from satisfactory. This leaves room for exploration of bootstrapping and augmentation methods or more sophisticated modeling to improve results.

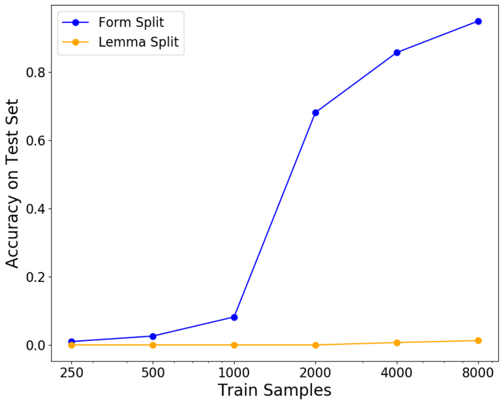


Figure 1: Inflection accuracy over *form-split* and *lemma-split* test sets as a function of train set size.

B Tech-Spec

All algorithms described in the paper were executed on a single machine equipped with one NVIDIA TITAN Xp GPU, 16 Intel i7-6900K(3.20GHz) CPUs and 126GB RAM. Since the LSTM algorithm was implemented on DyNet, there was no need of the GPU, and all the calculations were done using only the CPU.

C Hyper Parameters

1. Embedding size = 100
2. Hidden state size = 100
3. Attention size = 100
4. Number of LSTM layers = 1

During training, we experimented with several values for the hyper-parameters detailed above. However, for all the combinations we tried, the results barely changed both at the form-split setting and the lemma-split setting.

DQ-BART: Efficient Sequence-to-Sequence Model via Joint Distillation and Quantization

Zheng Li[†§1] **Zijian Wang**[§2] **Ming Tan**[2] **Ramesh Nallapati**[2] **Parminder Bhatia**[2]
Andrew Arnold[2] **Bing Xiang**[2] **Dan Roth**[2,3]
[1]Cornell University [2]AWS AI Labs [3] University of Pennsylvania
zl634@cornell.edu {zijwan, mingtan}@amazon.com
{rnallapa, parmib, anarnld, bxiang, drot}@amazon.com

Abstract

Large-scale pre-trained sequence-to-sequence models like BART and T5 achieve state-of-the-art performance on many generative NLP tasks. However, such models pose a great challenge in resource-constrained scenarios owing to their large memory requirements and high latency. To alleviate this issue, we propose to jointly distill and quantize the model, where knowledge is transferred from the full-precision teacher model to the quantized and distilled low-precision student model. Empirical analyses show that, despite the challenging nature of generative tasks, we were able to achieve a 16.5x model footprint compression ratio with little performance drop relative to the full-precision counterparts on multiple summarization and QA datasets. We further pushed the limit of compression ratio to 27.7x and presented the performance-efficiency trade-off for generative tasks using pre-trained models. To the best of our knowledge, this is the first work aiming to effectively distill and quantize sequence-to-sequence pre-trained models for language generation tasks.

1 Introduction

Pretrained sequence-to-sequence (seq2seq) models such as BART (Lewis et al., 2020; Liu et al., 2020) and T5 (Raffel et al., 2020; Xue et al., 2021) have shown great success in various natural language processing (NLP) tasks, such as text summarization (Nallapati et al., 2016; See et al., 2017; Narayan et al., 2018), machine translation, question answering (Fan et al., 2019) and information extraction (Zhou et al., 2021). However, such large-scale pre-trained language models come with hundreds of millions of parameters: Lewis et al. (2020) trained a BART model with 400M parameters, while Raffel et al. (2020) pushed the limit to 11 billion parameters in T5.

The continual growth in model sizes leads to significant demand in both computation and memory resources during inference, and poses a huge challenge on deployment, especially in real-time and/or resource-constrained scenarios. This motivates researchers to compress large pre-trained models to be smaller and faster while retaining strong performance. Among existing compression approaches such as weight-sharing (Dehghani et al., 2019; Lan et al., 2020), low-rank approximation (Ma et al., 2019; Lan et al., 2020), and pruning (Michel et al., 2019), quantization approaches have received attention recently since they reduce model footprints using lower bits for the weight values without changing the carefully-designed model architecture. Most prior work on transformer quantization focused on BERT-based transformers (Zhang et al., 2020; Zafrir et al., 2019; Bai et al., 2021). However, efficient quantization on the encoder-decoder transformers is insufficiently studied. Prato et al. (2020) achieve 8-bit quantization for a seq2seq transformer without significant loss of performance but low-bit quantization proved to be difficult for this model (4-bit performance in Table 2 in their work) due to the accumulation of quantization errors in seq2seq models. Moreover, their work did not target quantizing large-scale pre-trained language models, nor could it be applied to other NLP tasks besides machine translation. Meanwhile, model distillation which transfers knowledge from a large teacher model to a smaller student model has been widely investigated for BERT compression (Sanh et al., 2019; Jiao et al., 2020).

Recently, Shleifer and Rush (2020) applied "shrink and fine-tune" distillation method on BART for text summarization, yet their work focuses more on the methodology for distilling text summarization only. Besides, their work did not yield a sig-

[†]Work done during an internship at AWS AI Labs.
[§]Equal contribution.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 203 - 211
May 22-27, 2022 ©2022 Association for Computational Linguistics

nificant model footprint reduction, one of the most challenging issues in the deployment of large models in resource-constrained scenarios.

In this work, we try to address the challenge of building a more efficient seq2seq model by answering two research questions: first, how well does the quantized seq2seq model perform on various tasks? Second, how do we combine quantization and distillation to push the limit of compressing the seq2seq model without significant performance losses in challenging tasks like summarization and question answering? To this end, we proposed a joint distillation and quantization framework, which efficiently transfers the knowledge from a full-precision teacher seq2seq model to its student with fewer layers and ultra-low bits for encoding its parameters. Experimental results on BART show that the proposed models reduce the model footprint by 16.5x while preserving competitive performances on multiple language generation benchmarks, and further illustrate the performance-efficiency trade-off of compressing seq2seq models up to 27.7x smaller. To the best of our knowledge, this is the first work aiming to effectively distill and quantize seq2seq pre-trained models for language generation tasks.

2 Distilling and Quantizing BART

In this section, we consider two directions for reducing the size of our generative language model: quantization (§2.1) and distillation (§2.2). We apply distillation-aware training (§2.3) to train a quantized and distilled low-precision model as a student model to emulate the full-precision teacher model.

2.1 Quantization

Quantization refers to the operation of mapping a real (high-precision) number to its low-precision counterpart in order to achieve model footprint reduction. There has been extensive study on applying quantization to training neural networks. Different quantization schemes include, e.g., linear quantization (e.g., Hubara et al., 2016, 2017; Jacob et al., 2018), non-linear quantization (Li and Sa, 2019), approximation-based quantization method (Lin et al., 2016), and loss-aware quantization (Hou and Kwok, 2018). In our work, we used the approximation-based method with linear quantization following Zhang et al. (2020).

Quantizing BART We applied quantization to the weights of all the hidden layers and most of the embeddings. Following previous work (Zhang et al., 2020), we did not quantize positional embeddings and quantized activations only to 8 bits.

Weight Quantization We dive into the mathematical details of how to quantize the weights in BART models. Let us denote $\mathbf{w}^t \in \mathcal{R}^{n_t}$ as the vector obtained by stacking all the columns of the full-precision weight matrix $\mathbf{W}^t$ that we wish to quantize at iteration t. By quantizing $\mathbf{w}^t$, we are looking for a scaling factor (also known as quantization step) α^t and a low-precision number $\mathbf{b}^t$, to replace full precision weight $\mathbf{w}^t$ with $\alpha^t\mathbf{b}^t$. When quantizing with more than 2 bits, we are applying the commonly used symmetric linear quantization, with

$$\begin{aligned}\alpha^t &= \max_i |w_i^t| \,/\, th\\ \mathbf{b}^t &\in \{-th, \cdots, -1, 0, 1, \cdots, th\}^{n_t}\end{aligned}$$

where $th = 2^{n_b-1} - 1$ and n_b is the number of bits we use for quantization. Then $\mathbf{b}^t$ can be obtained by $\mathbf{b}^t = round(\mathbf{w}^t/\alpha^t)$. When quantizing with 2 bits, we use the approximation based TWN method (Li et al., 2016). The mathematical details are provided in Appendix A.

2.2 Distillation

The second task we consider is knowledge distillation, where we train a smaller student model to mimic the behavior of a larger teacher model; specifically, we want to reproduce the output logits, attentions, and hidden states of the teacher model. Following Shleifer and Rush (2020), we initialize the student model by copying the weights from maximally spaced layers of the teacher model, e.g., when initializing a 3-layer student encoder (decoder) from a 6-layer teacher encoder (decoder), we copy the 0th, 3th and 5th layers from the teacher to the student. When copying only 1 layer, we choose the last instead of the first, which has been shown empirically to yield better performance. Different than Shleifer and Rush (2020) who only distill the decoder, we distill both the encoder and the decoder. After initialization, we fine-tune the student model with the combined objective of task loss and distillation loss, i.e. $\mathcal{L}_{\text{data}} + \mathcal{L}_{\text{dist}}$, with

$$\mathcal{L}_{\text{dist}} = \mathcal{L}_{\text{logits}} + \mathcal{L}_{\text{att}} + \mathcal{L}_{\text{hid}}$$

where the RHS are MSE losses measuring the difference between the student and teacher with regard to output logits, attention scores (including

		Summarization						Long-form QA		
Model	Size (MB)	CNN/DailyMail			XSUM			ELI5		
W-E-A (#bits) E-D (#layers)		R1	R2	RL	R1	R2	RL	R1	R2	RL
32-32-32 6-6	531 (1x)	44.90	22.25	42.09	43.84	20.79	35.71	26.02	5.11	15.36
8-8-8 6-6 (direct quant.)	137 (3.9x)	11.36	1.01	11.01	22.74	5.69	17.81	6.72	0.43	4.89
		Distillation-Aware Quantization								
8-8-8 6-6	137 (3.9x)	44.66	21.92	41.86	42.51	19.61	34.61	27.10	5.15	16.23
2-2-8 6-6	39 (13.6x)	42.94	20.07	40.13	40.06	17.34	32.46	26.33	4.97	16.15
		Distillation-Aware Quantization + Distillation								
8-8-8 6-3	110 (4.8x)	43.99	21.25	41.24	41.94	19.21	34.21	26.38	5.13	16.27
8-8-8 6-1	92 (5.8x)	42.52	20.04	40.05	39.42	17.70	32.69	24.27	4.74	15.71
8-8-8 3-1	72 (7.4x)	41.18	18.75	38.58	36.39	15.29	29.91	23.69	453	15.51
2-2-8 6-3	32 (16.5x)	42.49	19.71	39.70	39.66	17.26	32.33	25.41	4.83	15.94
2-2-8 6-1	27 (19.2x)	41.14	18.72	38.66	36.61	15.33	30.22	23.34	4.31	15.20
2-2-8 3-1	22 (23.5x)	40.14	17.75	37.60	33.56	13.05	27.48	22.60	3.99	14.95
2-2-8 1-1	19 (27.7x)	39.00	16.73	36.42	29.04	9.56	23.47	21.51	3.44	14.30

Table 1: Distillation and quantization results on BART for text summarization on CNN/DailyMail and XSUM benchmarks and long-form question answering on the ELI5 benchmark. We abbreviate the number of bits for **w**eights, word **e**mbedding and **a**ctivations as "W-E-A (#bits)", followed by the number of **e**ncoder and **d**ecoder layers as "E-D (#layers)". We use the rouge-{1,2,L} as evaluation metrics (Lin, 2004). We found that distillation-aware quantized models achieves comparable or even better performance compared with the full precision models, and combining quantization and distillation, e.g., from "2-2-8 6-6" to "2-2-8 6-3", gives us a further boost in model footprint compression ratio without significant sacrifice in performance. See §3.2 for details.

encoder attention, decoder attention and cross attention), and hidden states (including all encoder and decoder layers).[1] We include the details of the loss in Appendix B for completeness.

2.3 Distillation-aware quantization

To fine-tune our quantized and distilled model, we use the technique of distillation-aware quantization with a teacher-student architecture from (Zhang et al., 2020)[2]. We treat the quantized and distilled low-precision model as a student model trained to emulate the full precision model, which in this case is the teacher model. Meanwhile, we also keep the full-precision distilled counterpart of the student model for parameter update. At each iteration, we first quantize the full precision student model to get its quantized version, then do the forward pass with the low-precision student model and get the task loss as well as the distillation losses discussed in §2.2. Finally, we use these losses to update the parameters in the full-precision student model.

3 Experiments and Discussions

In this section, we evaluate the efficacy of jointly Distilling and Quantizing BART (hereinafter, DQ-BART) on text summarization and long-form question answering using three benchmarks: CNN/DailyMail (See et al., 2017), XSUM (Narayan et al., 2018), and ELI5 (Fan et al., 2019). We additionally study machine translation with mBART on WMT English-Romanian (En-Ro) (Bojar et al., 2016).

3.1 Experimental Setup

We followed the standard splits of these datasets. The statistics could be found in Appendix C. For ELI5, we reproduced the author's implementation to train a dense retriever that retrieves 10 supporting documents from Wikipedia for each question. Additional details could be found in Appendix D.

As our target is achieving efficient seq2seq generative models, we used base-sized BART for summarization and question answering tasks. For machine translation, we used mBART-large due to the lack of pretrained base-sized multilingual BART models. We reused existing models[3], and finetuned our own models on end tasks when no open-sourced model is available. We trained our quantized-only models for 10 epochs and distilled-and-quantized

[1] Based on an initial small-scale study, we didn't find a significant difference between weighted and unweighted losses in our setting. For simplicity, we use unweighted loss here and leave the tuning of weights for future work.

[2] Note that in this work we jointly distill and quantize encoder-decoder models, while Zhang et al. (2020) used a similar technique but 1) for quantizing encoder-only models and 2) without the actual model distillation.

[3] https://huggingface.co/ainize/bart-base-cnn; https://huggingface.co/facebook/mbart-large-en-ro

models for 20 epochs. We used a batch size of 128, a learning rate of 3×10^{-5} with 5% linear warmup, and selected the best model based on rouge-L scores on the development set. We set generative hyperparameters following previous work (Lewis et al., 2020). All experiments were performed on A100 GPUs.

3.2 DQ-BART Results and Discussions

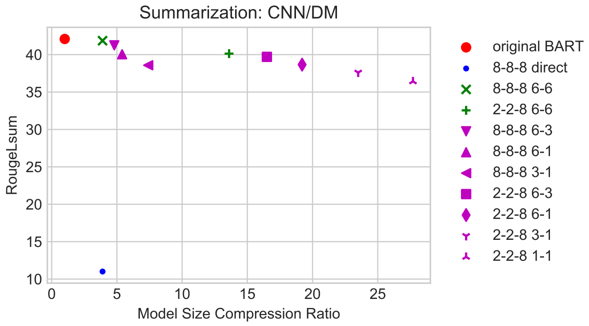


Figure 1: Visualization of performance v.s. model footprint compression ratio on CNN/DailyMail based on Table 1. Green dots are for quantization only, and purple dots are for distillation + quantization. We found that the performance degradation is minimal as the compression ratio grows, especially before 20x.

We summarized the main results in Table 1 and visualized the performance on text summarization on the CNN/DailyMail dataset in Figure 1. Additional visualizations are in Appendix E. We found that:

1. Direct quantization performs poorly in generation tasks. The rouge-L score drops ∼50-75% relatively compared with the baseline.
2. The performance of 8-bit distillation-aware quantized models ("8-8-8 6-6") achieves comparable or even better performance compared with the full precision models across all tasks, signaling that 8-bit is not too challenging for generative models like BART, similar to the findings for BERT (Zhang et al., 2020).
3. We were able to achieve a 13.6x model size compression ratio when using 2-bit quantization with the trade-off of slight performance drop for summarization tasks and even no performance drop for the long-form QA task.
4. Combining quantization and distillation gives us a further boost in model compression ratio without significant further sacrifice in performance. For example, when using 2-bit quantization, by cutting the layers of the decoder in half (from "2-2-8 6-6" to "2-2-8 6-3"), we only saw $<$ 0.5 rouge-L performance drop across all tasks while getting another 2.9x compression.
5. When pushing the compression rate to the limit ("2-2-8 1-1"), we were able to achieve a 27.7x compression ratio while still preserving reasonable performance. We observed a rouge-L drop of 5.67 for CNN/DailyMail ($42.09 \rightarrow 36.42$), 12.24 for XSUM ($35.71 \rightarrow 23.47$), and 1.06 for ELI5 ($15.36 \rightarrow 14.30$). Thus, for certain tasks a large model compression ratio would not lead to a significant performance drop while for others the drop could be huge, suggesting that the specific compression ratio to use should be decided on a task-by-task basis with the trade-off of performance and efficiency in mind.

3.3 DQ-mBART for Translation

We further extend our study to see how distillation and quantization work for mBART (Liu et al., 2020), a deeper multilingual model. We experimented mBART-large on WMT English-Romanian translation task (Bojar et al., 2016). The results are in Table 2.

Model	Size	BLEU
32-32-32 12-12	1x	26.82
8-8-8 12-12 (direct quant.)	4.0x	0.01
Distillation-Aware Quantization		
8-8-8 12-12	4.0x	25.91
2-2-8 12-12	15.2x	23.48
Distillation-Aware Quantization + Distillation		
8-8-8 12-6	4.7x	25.61
8-8-8 12-3	5.2x	24.22
8-8-8 12-1	5.6x	20.61
2-2-8 12-6	18.0x	17.66
2-2-8 12-3	19.9x	16.99
2-2-8 12-1	21.3x	12.81
2-2-8 1-1	30.6x	10.36

Table 2: Distillation and quantization results for translation on WMT16 En-Ro with mBART-large.

We found that distillation-aware quantization yields reasonably good performance, similar to the findings in DQ-BART (Table 1). However, the performance drops substantially when performing 2-bit quantization with distillation, possibly due to the accumulation of the distillation/quantization error becoming more significant with deeper models and the challenging nature of machine translation.

Future work may explore how to improve the performance of joint distillation and quantization for deep models under a low-bit setting.

3.4 Distillation and Quantization v.s. Distillation Only

	Model	Size	R1	R2	RL
CNN/DM	Distill + Quant 8-8-8 6-3	**4.8x**	**43.99**	**21.25**	**41.24**
	Distill only 16-16-16 3-1	3.8x	41.17	18.72	38.62
XSUM	Distill + Quant 8-8-8 6-3	**4.8x**	**41.94**	**19.21**	**34.21**
	Distill only 16-16-16 3-1	3.8x	36.60	15.46	30.07
ELI5	Distill + Quant 8-8-8 6-3	**4.8x**	**26.38**	**5.13**	**16.27**
	Distill only 16-16-16 3-1	3.8x	23.80	4.54	15.40

Table 3: Comparisons between distillation-only and joint distillation and quantization.

We want to understand how much gain there is when doing joint distillation and quantization compared with distillation-only method (Shleifer and Rush, 2020). To do so, we trained distillation-only models and compared them with DQ-BART with a similar size. From Table 3, we found that joint distillation and quantization performs much better across all tasks, signaling the huge gain with joint distillation and quantization. Additional ablation study on "Shrink and Finetune" could be found in Appendix F.

4 Conclusion

Transformer-based pre-trained seq2seq language models like BART have greatly advanced the state of the art in a range of NLP tasks. Yet, these extremely large-scale models pose a challenge in resource-constrained scenarios. To alleviate this issue, we proposed DQ-BART, a jointly distilled and quantized BART model. Empirical results show that, despite the difficult nature of language generation tasks, we achieve a 16.5x model footprint compression ratio with little performance drop on three generative benchmarks, and further present the performance-efficiency trade-off for seq2seq models up to a 27.7x compression ratio. Additionally, we studied distillation and quantization for mBART on a machine translation task, and highlighted the challenge of joint low-bit quantization with distillation for deeper models on cross-lingual tasks. To the best of our knowledge, our method is the first to apply joint quantization and distillation on pretrained language models, and this is the first work aiming to effectively distill and quantize seq2seq pretrained models for language generation tasks. We hope this work could open doors for developing and applying efficient seq2seq language models. We leave additional compression methods like attention head pruning (Michel et al., 2019) and sequence-level distillation (Kim and Rush, 2016), and the measurement of latency improvements in various settings for future work. Our code is available at `https://www.github.com/amazon-research/dq-bart/`.

Acknowledgment

We would like to thank colleagues at AWS AI Labs and our anonymous ARR reviewers for their constructive feedback.

References

Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jin Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. 2021. BinaryBERT: Pushing the limit of BERT quantization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4334–4348, Online. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany. Association for Computational Linguistics.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. 2019. Universal transformers. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Lu Hou and James T. Kwok. 2018. Loss-aware weight quantization of deep networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2016. Binarized neural networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4107–4115.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2017. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, 18(1):6869–6898.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew G. Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2704–2713. IEEE Computer Society.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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

Fengfu Li, Bo Zhang, and Bin Liu. 2016. Ternary weight networks. In *Proceedings of 1st International NIPS Workshop on EMDNN*.

Zheng Li and Christopher De Sa. 2019. Dimension-free bounds for low-precision training. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11728–11738.

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

Zhouhan Lin, Matthieu Courbariaux, Roland Memisevic, and Yoshua Bengio. 2016. Neural networks with few multiplications. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

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

Xindian Ma, Peng Zhang, Shuai Zhang, Nan Duan, Yuexian Hou, Ming Zhou, and Dawei Song. 2019. A tensorized transformer for language modeling. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 2229–2239.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14014–14024.

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

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

Gabriele Prato, Ella Charlaix, and Mehdi Rezagholizadeh. 2020. Fully quantized transformer for machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1–14, Online. Association for Computational Linguistics.

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

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv preprint*, abs/1910.01108.

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

Sam Shleifer and Alexander M Rush. 2020. Pre-trained summarization distillation. *ArXiv preprint*, abs/2010.13002.

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

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pages 36–39. IEEE.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020. TernaryBERT: Distillation-aware ultra-low bit BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521, Online. Association for Computational Linguistics.

Ben Zhou, Kyle Richardson, Qiang Ning, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2021. Temporal reasoning on implicit events from distant supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1361–1371, Online. Association for Computational Linguistics.

A Details of TWN Quantization

When quantizing using 2 bits (which is also know as ternarization), following Zhang et al. (2020), we apply the TWN method (Li et al., 2016). To quantize $\mathbf{w}$, we are looking for scaling factor $\alpha > 0$ and $\mathbf{b} \in \{-1, 0, 1\}^n$ such that $\mathbf{w} \sim \alpha\mathbf{b}$ where n is the dimension of $\mathbf{w}$. To minimize the quantization error, we have the following optimization problem:

$$\alpha^*, \mathbf{b}^* = \arg\max_{\alpha,\mathbf{b}} ||\mathbf{w} - \alpha\mathbf{b}||^2$$
$$\text{where } \alpha > 0, \mathbf{b} \in \{-1, 0, 1\}^{dim(\mathbf{w})}$$

Denote Δ as a threshold and $I_\Delta(x)$ be a function such that

$$I_\Delta(x) = \begin{cases} 1, \text{ if } x > \Delta \\ 0, \text{ if } -\Delta \le x \le \Delta \\ -1, \text{ if } x < -\Delta \end{cases}$$

and denote set $J_\Delta = \{i \mid I_\Delta(\mathbf{w}_i) \neq 0\}$, then according to Hou and Kwok (2018), the solution to the previous optimization problem can be reached at

$$\mathbf{b}^* = I_{\Delta^*}(\mathbf{w}), \alpha^* = \frac{||\mathbf{w} \odot \mathbf{b}^*||_1}{||\mathbf{b}^*||_1},$$

$$\text{with } \Delta^* = \arg\max_\Delta \frac{1}{|J_\Delta|}\left(\sum_{i \in J_\Delta} |\mathbf{w}_i|\right)$$

where $\odot$ is element-wise multiplication and $||\cdot||_1$ is the l_1-norm. To approximate this result, we set $\Delta^* = 0.7||\mathbf{w}||_1/dim(\mathbf{w})$ then compute α^* and $\mathbf{b}^*$ accordingly.

B Details of Distillation Losses

The distillation losses is defined as the following:

$$\mathcal{L}_{\text{dist}} = \mathcal{L}_{\text{logits}} + \mathcal{L}_{\text{att}} + \mathcal{L}_{\text{hid}}$$

In this section we'll go through each part of the losses. We denote $\phi_{enc}(\cdot), \phi_{dec}(\cdot)$ as the functions that map the index of an encoder/decoder layer of the student model to the index of the teacher model layer that it is trained to emulate, the details of which is discussed in §2.2, and we use l^S_{enc}, l^S_{dec} to denote the number of encoder layers and decoder layers of the student model. To illustrate, if $l^S_{enc} = 3, l^S_{dec} = 2$, we would have:

$$\phi_{enc}(0, 1, 2) = 0, 3, 5, \quad \phi_{dec}(0, 1) = 0, 5$$

For simplicity, we use superscript $\cdot^S, \cdot^T$ to distinguish counterparts from the student model and teacher model respectively.

Next, we will explain the definition of each part of the distillation losses.

Firstly, $\mathcal{L}_{\text{logits}}$ is the Mean Squared Error (MSE) between the output logits of the student model and that of the teacher model, i.e.

$$\mathcal{L}_{\text{logits}} = MSE(logits^S, logits^T)$$

Secondly, $\mathcal{L}_{\text{att}}$ is the attention distillation loss, which is the sum of distillation losses of encoder attentions (EA), decoder attentions (DA), and cross attention (CA), i.e.

$$\mathcal{L}_{\text{att}} = \mathcal{L}_{\text{EA}} + \mathcal{L}_{\text{DA}} + \mathcal{L}_{\text{CA}}$$

where

$$\mathcal{L}_{\text{EA}} = \sum_{i=1}^{l^S_{enc}} MSE(EA^S_i, EA^T_{\phi_{enc}(i)})$$

$$\mathcal{L}_{\text{DA}} = \sum_{i=1}^{l^S_{dec}} MSE(DA^S_i, DA^T_{\phi_{dec}(i)})$$

$$\mathcal{L}_{\text{CA}} = \sum_{i=1}^{l^S_{dec}} MSE(CA^S_i, CA^T_{\phi_{dec}(i)})$$

with the subscripts $i, \phi(i)$ specifying the indices of the layers.

Finally, $\mathcal{L}_{\text{hid}}$ is the distillation loss between all the hidden states between student layers and teacher layers, which include encoder hidden states (EHS) and decoder hidden states (DHS):

$$\mathcal{L}_{\text{hid}} = \mathcal{L}_{\text{EHS}} + \mathcal{L}_{\text{DHS}}$$

where

$$\mathcal{L}_{\text{EHS}} = \sum_{i=1}^{l^S_{enc}} MSE(EHS^S_i, EHS^T_{\phi_{enc}(i)})$$

$$\mathcal{L}_{\text{DHS}} = \sum_{i=1}^{l^S_{dec}} MSE(DHS^S_i, DHS^T_{\phi_{dec}(i)})$$

C Dataset Statistics

Dataset	Dataset Split Count			Mean Token Length	
	Train	Valid.	Test	Source	Target
CNN/DM	87,113	13,368	11,490	691	52
XSUM	204,045	11,332	11,334	374	21
ELI5	272,634	9,812	24,512	Q: 38 D: 1,672	111
WMT16 En-Ro	610,320	1,999	1,999	21	21

Table 4: Dataset Statistics.

D ELI5 Additional Details

In this section, we present additional details for the ELI5 dataset.

D.1 Dense Retriever

We were not able to find a public version of supporting documents for ELI5, and thus followed the author's implementation[4] to train a dense retriever that retrieves support documents from Wikipedia. Our trained retriever achieves a similar performance compared with the one reported in the author's implementation (recall: ours 0.3273, reported 0.3247).

D.2 Evaluating ELI5 Results

We use the ROUGE-SCORE package[5] to calculate rouge scores through the paper. However, as the author of ELI5 pointed out[4], the original rouge implementation used in ELI5 and BART papers performs additional normalization. For consistency, we also reported results for ELI5 using the same ROUGE-SCORE package, which differs from the one used in ELI5/BART. Here we compared the performance of our trained ELI5 baseline model with the public one using the rouge implementation used in ELI5/BART papers.

Model	Rouge Setting	R1	R2	RL
BART-base (ours)	rouge-score	26.02	5.11	15.36
	BART/ELI5	29.19	5.59	25.88
BART-large reported (Lewis et al., 2020)	BART/ELI5	30.60	6.20	24.30

Table 5: Comparison when using different rouge implementation.

Results in Table 5 shows that the performance of our base-size model is close to the one with large-size reported in Lewis et al. (2020). This signals that our baseline model for ELI5 is well-trained.

E Visualizations of Experimental Results on XSUM and ELI5 datasets

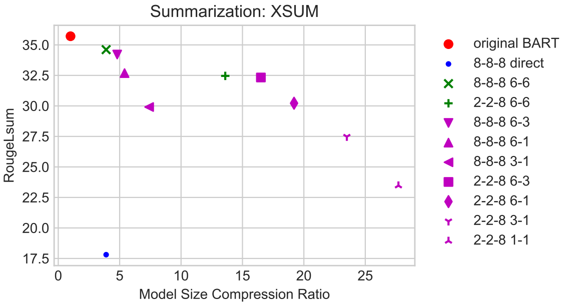


Figure 2: Visualization of performance v.s. model footprint compression ratio on XSUM based on Table 1.

[4] https://yjernite.github.io/lfqa.html

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

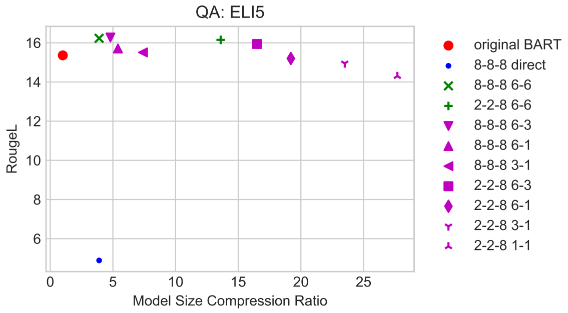


Figure 3: Visualization of performance v.s. model footprint compression ratio on ELI5 based on Table 1.

F Comparisons on "Shrink and Finetune"

We benchmarked the performance of three randomly picked models with the "Shrink and Finetune" schema proposed in Shleifer and Rush (2020). We ran the models using the same hyperparameter settings we used in this paper. The results are shown in Table 6.

We found that when using distillation losses between the teacher and the student, the performance are slightly better than the "Shrink and Finetune" method under our setting. This signals that having guidance in weighting is important for a quantized and distilled model to learn well.

Model	Loss	R1	R2	RL
CNN/DM	Ours	**42.52**	**20.04**	**40.05**
8-8-8 6-1	S&F	42.29	19.92	39.83
XSUM	Ours	**40.06**	**17.34**	**32.46**
2-2-8 6-6	S&F	39.69	17.27	32.27
ELI5	Ours	**27.10**	**5.15**	**16.23**
8-8-8 6-6	S&F	26.95	5.10	16.16

Table 6: Performance comparison between the loss used in this paper and the "shrink and finetune" loss from (Shleifer and Rush, 2020).

Learning-by-Narrating: Narrative Pre-Training for Zero-Shot Dialogue Comprehension

Chao Zhao[1]* Wenlin Yao[2] Dian Yu[2]
Kaiqiang Song[2] Dong Yu[2] Jianshu Chen[2]
zhaochao@cs.unc.edu
{wenlinyao,yudian,riversong,dyu,jianshuchen}@tencent.com
[1] UNC Chapel Hill, Chapel Hill, NC [2] Tencent AI Lab, Bellevue, WA

Abstract

Comprehending a dialogue requires a model to capture diverse kinds of key information in the utterances, which are either scattered around or implicitly implied in different turns of conversations. Therefore, dialogue comprehension requires diverse capabilities such as paraphrasing, summarizing, and commonsense reasoning. Towards the objective of pre-training a zero-shot dialogue comprehension model, we develop a novel narrative-guided pre-training strategy that *learns by narrating* the key information from a dialogue input. However, the dialogue-narrative parallel corpus for such a pre-training strategy is currently unavailable. For this reason, we first construct a dialogue-narrative parallel corpus by automatically aligning movie subtitles and their synopses. We then pre-train a BART model on the data and evaluate its performance on four dialogue-based tasks that require comprehension. Experimental results show that our model not only achieves superior zero-shot performance but also exhibits stronger fine-grained dialogue comprehension capabilities. The data and code are available at https://github.com/zhaochaocs/Diana.

1 Introduction

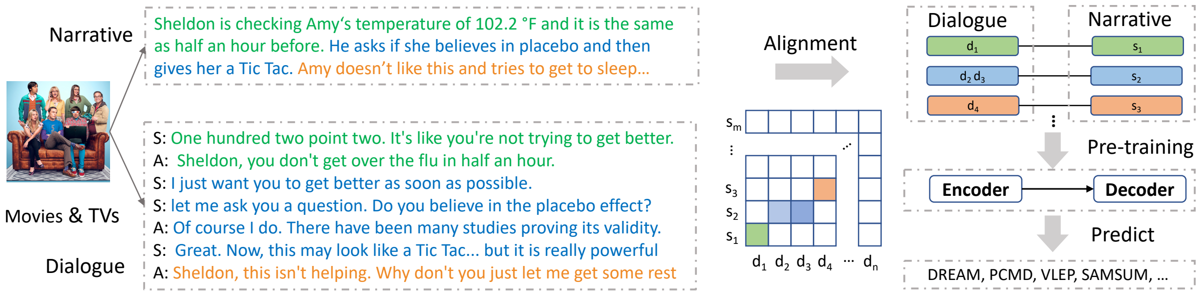
Dialogue comprehension (Sun et al., 2019; Cui et al., 2020) aims to capture diverse kinds of key information in utterances, which are either scattered around or implicitly implied in different turns of conversations. Therefore, it requires different capabilities such as paraphrasing (Falke et al., 2020), summarizing (Gliwa et al., 2019), and commonsense reasoning (Arabshahi et al., 2021). Recent advances in pre-trained language models (PLMs) (Devlin et al., 2019; Radford et al., 2019) have been applied to the problem (Jin et al., 2020; Liu et al., 2021). However, these PLMs are generally pre-trained on formal-written texts, which are different from dialogue data in nature. Specifically, dialogues are composed of colloquial languages from multi-speakers, and utterances usually have complex discourse structures (Afantenos et al., 2015). Therefore, applying these models directly to dialogue comprehension, especially in low-resource settings, is sub-optimal.

To learn better dialogue representations, recent studies have designed several dialogue-specific pre-training objectives such as speaker prediction (Qiu et al., 2021), utterance prediction (Chapuis et al., 2020), response selection (Wu et al., 2020), and turn order restoration (Zhang and Zhao, 2021). These methods, albeit improve over the vanilla PLMs, usually rely on surface-level dialogue information. In particular, they still fail to train the models to explicitly learn the aforementioned capabilities which are critical for dialogue comprehension (e.g., linguistic knowledge, world knowledge, and commonsense knowledge). Furthermore, it was not able to incorporate knowledge beyond dialogue (e.g., non-verbal communications between speakers, as well as time and location information), which are also crucial for dialogue comprehension.

To pre-train a zero-shot dialogue comprehension model with the aforementioned features, we develop a novel generative pre-training strategy that *learns by narrating* the key information from a dialogue input (see Figure 1 for an example). In particular, the generated narrative text is supposed to not only (i) paraphrase the gists of the dialogue but also (ii) carry certain inferred information (e.g., the time and location of a scene and relations between speakers) that are not explicitly mentioned in the dialogues. Learning to narrate such information helps the model to learn varied lexical, syntactic, and semantic knowledge of dialogue. It also enhances the model's ability to infer extra information beyond the literal meaning within dialogues, which will benefit the model's capability of dialogue comprehension.

[*] Work was done during the internship at Tencent AI lab.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 212 - 218
May 22-27, 2022 ©2022 Association for Computational Linguistics



Sheldon is checking Amy's temperature of 102.2 °F and it is the same as half an hour before. He asks if she believes in placebo and then gives her a Tic Tac. Amy doesn't like this and tries to get to sleep...

S: One hundred two point two. It's like you're not trying to get better.
A: Sheldon, you don't get over the flu in half an hour.
S: I just want you to get better as soon as possible.
S: let me ask you a question. Do you believe in the placebo effect?
A: Of course I do. There have been many studies proving its validity.
S: Great. Now, this may look like a Tic Tac... but it is really powerful
A: Sheldon, this isn't helping. Why don't you just let me get some rest

Figure 1: Overview of the *learning-by-narrating* strategy for pre-training a zero-shot dialogue comprehension model (with an encoder-decoder architecture).

However, the *learning-by-narrating* strategy would require a dialogue-narrative parallel corpus, which, to our best knowledge, is not publicly available. For this reason, we first create **DIANA**, a large-scale dataset with (**DIA**logue, **NA**rrative) pairs automatically collected from subtitles of movies and their corresponding plot synopses. We consider dialogues from movie subtitles as they are close to daily human-to-human conversations (Zhang and Zhou, 2019). In addition, the movie synopses include rich narrative information, which is helpful for dialogue comprehension. After data collection and strict quality control, we obtain a dataset with 243K (dialogue, narrative) pairs written in English. As the automatic data construction procedure is language-independent, it can be applied to low-resource languages as well.

We then pre-train a BART model (Lewis et al., 2020) on the constructed corpus with the proposed *learning-by-narrating* strategy, and evaluate it on four dialogue-based tasks that require comprehension. In zero-shot settings, our pre-trained model outperforms the BART baseline on all tasks by a large margin (e.g., +8.3% on DREAM (Sun et al., 2019)), demonstrating the success of our approach.

The contributions of this paper are three-fold:

- We propose a novel *learning-by-narrating* pre-training strategy for dialogue comprehension;
- We release **DIANA**, a new large-scale dialogue-narrative parallel corpus;
- Experiments show that our pre-trained dialogue comprehension model achieves superior zero-shot performance on a variety of downstream tasks.

2 DIANA: A Dialogue-Narrative Corpus

In this section, we describe the procedure to create the dialogue-narrative parallel dataset.

2.1 Data Collection and Segmentation

We collect 47,050 English subtitles of movies and TV episodes released from Opensubtitle (Lison et al., 2018) and their corresponding synopses from online resources such as Wikipedia and TMDB. To link the subtitle and synopsis of the same movie or TV episode, we require a subtitle and a synopsis to have the same title and the release year, as well as a high overlap rate ($> 50\%$) on role names.

The subtitle and synopsis of a movie are too long for a PLM. To facilitate pre-training, we split both the subtitle and synopsis into smaller segments and align the related segments from each part to shorter (dialogue, narrative) pairs. We split subtitles using the time interval δ_T between utterances and split a synopsis into sentences. We set $\delta_T = 5s$.

2.2 Data Alignment

We aim to align the dialogue sessions $\{d_1, \dots, d_n\}$ and narrative segments $\{s_1, \dots, s_m\}$ with maximum global similarity to form (dialogue, narrative) pairs. For each dialogue session d_j, the goal is to find its corresponding narrative segment s_i.

Inspired by (Tapaswi et al., 2015) in which the narrative in a synopsis follows the timeline of a movie or a TV episode, we develop a dynamic time warping method to find the globally optimal alignment score. During aligning, some narrative segments contain information beyond the dialogue, so they cannot be aligned to any dialogue session. We therefore allow our algorithm to skip at most k narrative segments during alignment searching:

$$\mathcal{A}(i,j) = \max_{0 \leq k \leq K+1} \mathcal{A}(i-k, j-1) + \mathcal{S}\left(s_i, d_j\right), \quad (1)$$

where $\mathcal{A}(i,j)$ denotes the optimal alignment score of the first i narrative segments and the first j dialogue sessions. $\mathcal{S}\left(s_i, d_j\right)$ is the text similarity between s_i and d_j.

We compare the performance of three text similarity measures: Jaccard similarity, Rouge-1F, and

Similarity Function	Accuracy
Jaccard	57.98
Rouge-1F	60.01
TF-IDF	67.20
TF-IDF normalized	**71.95**

Table 1: Alignment accuracy of different similarity measures on MovieNet.

TF-IDF. In consideration of time efficiency, we don't apply more advanced neural methods. We compare these similarity measures on MovieNet dataset (Huang et al., 2020), which provides a manual alignment between the segments of subtitles and synopses of 371 movies. [1] We evaluate the performance of each similarity measure by alignment accuracy, a.k.a, the percentage of dialogue sessions that are correctly aligned to the corresponding narrative segment. As shown in Table 1, TF-IDF performs best among all similarity measures. We also find that a narrative-wise L_2 normalization of the TF-IDF can further improve the alignment accuracy. It helps to penalize the similarity of (d_j, s_i) when s_i has high similarity with many dialogues (e.g., when s_i contains common words or protagonists' names.) We therefore choose the normalized TF-IDF as our similarity function. We further analyze the errors during alignment and find that 85.94% of errors happen because the dialogue session is aligned to the previous or next segment of the gold narrative segment. It indicates that most of the errors happen locally. Figure 2 shows an example from MovieNet, where the red line and the blue line indicate the gold alignment and the predicted alignment via normalized TF-IDF, respectively. It shows that the two lines are generally well overlapped except for some local discrepancies.

2.3 Quality Control

After data alignment, each narrative segment s_i can be aligned to multiple dialogues. To consider the local alignment errors, we also merge the aligned dialogues of s_{i-1} and s_{i+1} to the dialogues of s_i. Some of these dialogues may not be relevant to s_i. To select the relevant dialogues, we use a greedy method to incrementally select dialogues until the rouge-F score between the narrative and the selected dialogues doesn't increase. After selection, we concatenate the selected dialogues and preserve their relative position. We finally obtain around 1.5 Million (dialogue, narrative) pairs.

[1] We use MovieNet for test purposes only.

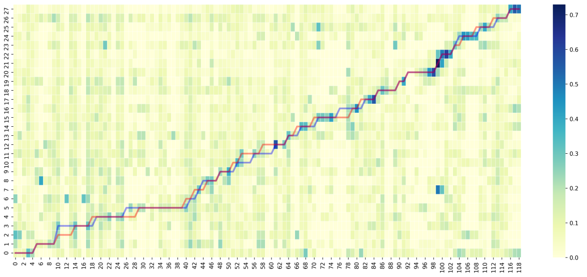


Figure 2: The Alignment of dialogues and narrative segments of a movie. X-axis and Y-axis are the ID of dialogue sessions and narrative segments, respectively. The variety of colors depicts the different similarity values between a dialogue session and a narrative segment. The blue line is the predicted alignment via normalized TF-IDF while the red line is the gold alignment.

To further improve the quality of data, we filter out pairs where the dialogue and the narrative are irrelevant. To evaluate the relevance, we use two automatic measures: Coverage and Density (Grusky et al., 2018). Low Coverage and Density indicate that the narrative text is either too abstractive or irrelevant to the dialogue. We thus only select the pairs with Coverage > 0.5 and Density > 1. After this strict quality control, we obtain 243K (dialogue, narrative) pairs as the final DIANA dataset, which is a high-quality subset of the original dataset. The average length of the dialogue and the narrative are 58 tokens and 18 tokens, respectively.

2.4 Analysis of Knowledge Type

To analyze what types of knowledge are included in DIANA, we randomly sample 100 instances and manually categorize the relation between dialogue and the corresponding narrative text into seven knowledge types. We show the percentage of each knowledge type in parentheses and in Figure 3 as well. The knowledge types are:

- **Summarizing** (39%): The narrative text summarizes multiple utterances as a concise statement to reflect the salient event or information of the dialogue.
- **Visual/Audial** (17%): The narrative text provides extra visual or audial information of the dialogue, such as the location of the dialogue, the speakers' actions, and ambient sounds.
- **Paraphrasing** (14%): The narrative text restates speakers' utterances using other words.
- **Text Matching** (9%): The narrative text is directly copied from the utterances of speakers.

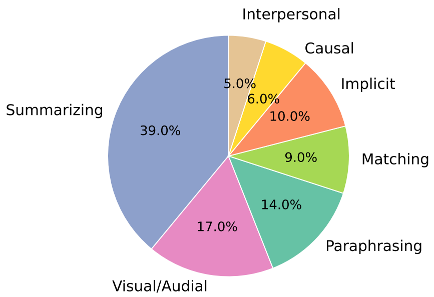


Figure 3: The knowledge type distribution in DIANA.

- **Implicit** (10%): The narrative text provides extra information that is not explicitly mentioned in the dialogue.
- **Causal** (6%): The narrative text describes the cause and effect relationship between events.
- **Interpersonal** (5%): The narrative text reveals the relationships between speakers.

Among these knowledge types, *Summarizing* and *Visual/Audial* are the two most frequent ones. They are followed by *Paraphrasing* and *Text Matching*, which contribute to 23% in total. It also shows that narratives use paraphrasing more often than copying. Additionally, DIANA contains three higher-level knowledge types that require the awareness of real-world commonsense and more complicated inference such as implicit knowledge, causal relationships, and interpersonal relationships. The diverse knowledge types in DIANA indicate the benefit of this dataset for dialogue comprehension and other downstream tasks as well.

3 Pre-training: Learning-by-Narrating

During pre-training, we aim to inject the knowledge contained in DIANA into pre-trained models. One option is to ask the model to distinguish between a correct narrative and an incorrect narrative via a classification objective. However, it requires carefully designing additional non-trivial negative (dialogue, narrative) pairs. Therefore, we propose to directly generate a narrative text from the given dialogue by maximizing the generative probability:

$$p(\mathbf{y} \mid \mathbf{x}; \boldsymbol{\theta}) = \prod_{t=1}^{|\mathbf{y}|} p\left(y_t \mid y_{1:t-1}, \mathbf{x}; \boldsymbol{\theta}\right), \quad (2)$$

where $\mathbf{x}$ are dialogue texts and $\mathbf{y}$ are narrative texts.

There are two main advantages of using the generative objective. First, it can fully leverage the narrative information from each token of the narrative text with no need to construct negative pairs. Second, the pre-trained model can be directly applied to both generative and discriminative downstream tasks without further fine-tuning. For discriminative tasks, we calculate the probability of each candidate according to Equation 2 and choose the most probable candidate as the predicted answer.

4 Experiments

In this section, we evaluate the performance of the pre-trained model on four downstream tasks that require dialogue comprehension.

4.1 Setting

We use BART, a state-of-the-art sequence-to-sequence model, as our baseline model.[2] We use its released checkpoint and further pre-train the model on DIANA. During pre-training, we concatenate the utterances as the input and update the parameters to maximize the probability of the corresponding narrative. We use Adam as the optimizer, and we set the learning rate and weight decay to $3{\times}10^{-5}$ and 0.01, respectively. Following previous studies that suggest that a larger batch size helps pre-training, we set the batch size to 1024 and pre-train the model for 1,000 steps.

4.2 Tasks

We evaluate our model's ability of dialogue comprehension on four downstream tasks. **DREAM** (Sun et al., 2019) aims to read a dialogue and select the correct answer from options of a dialogue-related question. To make the task similar to our pre-training task, we follow previous work (Chen et al., 2021) to train a T5 model to convert each (question, answer) pair to a statement. **PCMD** (Ma et al., 2018) is a passage completion task. Given a dialogue and a passage that describes the dialogue, a query is created by replacing a character mention with a variable x, and the model needs to recover the character mention. **VLEP** (Lei et al., 2020) aims to select the most probable future event given the dialogue of the current event and two candidates of future events. **SAMSum** (Gliwa et al., 2019) is a dialogue summarization task to create a concise abstractive summary for a dialogue. The first three are discriminative tasks, and SAMSum is a generative task. None of the source dialogues in these tasks are included in DIANA.

[2] We also tried T5 and Pegasus in our early experiments but did not observe better performance compared with BART.

	Data	Task	DREAM	PCMD	VLEP	SAMSum		
			ACC	ACC	ACC	R1	R2	RL
BART-FT	-	-	62.56	75.89	65.07	49.18	24.47	47.12
GPT-2	-	-	41.99	45.02	54.58	10.83	0.74	11.68
RoBERTa	-	-	45.22	46.25	52.28	-	-	-
BART	-	-	45.07	46.07	54.26	29.92	9.58	28.54
	DIAL	DE	46.69	47.34	55.98	30.08	9.52	29.36
	CNN	CLS	50.46	49.27	55.53	-	-	-
	CNN	GEN	52.72	45.34	58.13	31.33	9.08	28.03
	CRD3	GEN	52.96	45.71	57.12	27.07	9.09	27.64
Narrator	DIANA	GEN	**53.41**	**54.88**	**58.90**	**37.27**	**13.23**	**36.12**

Table 2: Results on four dialogue-based tasks. For models that require further pre-training, we list the corresponding pre-training dataset and task.

Question Type	BART	Narrator
Paraphrase+Matching	58.4	66.1 (+7.7)
Reasoning	42.2	46.2 (+4.0)
Summary	51.1	53.4 (+2.3)
Logic	43.8	48.2 (+4.4)
Commonsense	37.8	41.9 (+4.1)
Arithmetic	23.8	23.8 (+0.0)

Table 3: Accuracy by question types on DREAM.

We evaluate the model performance on these tasks under the zero-shot setting. For discriminative tasks, we convert each test instance with K answer candidates as K (dialogue, narrative) pairs. Given the dialogue as input, we evaluate the conditional probability of each narrative according to Equation 2 and choose the most probable narrative as the predicted answer. We use accuracy (ACC) as the evaluation metric for discriminative tasks and ROUGE for the summarization task.

We compare our pre-trained model (Narrator) with strong pre-trained baselines such as GPT-2, RoBERTa, and BART. To investigate the impact of the pre-training objective, we compare with 1) BART-DIAL-DE: the original BART de-noising objectives, which is trained on the dialogue part of DIANA; and 2) BART-CNN-CLS: a classification objective, which is trained using the CNNDM dataset (See et al., 2017) to distinguish between positive and negative summaries based on the documents. Negative summaries are obtained from DocNLI (Yin et al., 2021) by replacing the words, entities, and sentences of positive summaries. We also investigate the quality of DIANA by comparing it with two large summarization datasets: CNNDM and CRD3 (Rameshkumar and Bailey, 2020). We pre-train BART to generate the summaries of these datasets from the corresponding documents and refer to the models as BART-CNN-GEN and BART-CRD3-GEN. Besides the zero-shot models, we list the supervised results finetuned on BART (BART-FT) as a reference for the upper bound.

4.3 Results

Results are shown in Table 2. Our observations are as follows. (i) When compared with vanilla PLMs, Narrator outperforms GPT-2, RoBERTa, and BART, demonstrating that the learning-by-narrating pre-training objective can improve the model's ability of dialogue comprehension. (ii) When compared with different pre-training tasks, Narrator outperforms BART-DIAL-DE, and BART-CNN-GEN outperforms BART-CNN-CLS. This indicates that the narrative-guided generative objective is more effective than the de-noising objective and the discriminative objective. (iii) When compared with different pre-training data, Narrator achieves better performance on all tasks compared with BART-CNN-GEN and BART-CRD3-GEN, demonstrating that DIANA is a more helpful resource for dialogue comprehension.

We further analyze what types of knowledge are enhanced during pre-training. To this end, we test Narrator on a subset of the DREAM test set, which includes annotated knowledge types released along with the DREAM dataset. As shown in Table 3, compared with the vanilla BART, Narrator achieves better performance on all knowledge types except Arithmetic, which is not covered in DIANA. The performance gain indicates that the narrative pre-training contributes the most to the knowledge related to paraphrasing and matching. It also benefits from other knowledge types that require various reasoning abilities such as commonsense reasoning and logic reasoning.

5 Conclusion

We propose a *learning-by-narrating* strategy to pre-train a zero-shot dialogue comprehension model. We first construct a dialogue-narrative dataset named DIANA, which contains 243K (dialogue, narrative) pairs obtained by automatically aligning movie subtitles with their corresponding synopses. We then pre-train a dialogue comprehension model based on DIANA and evaluate its performance on four downstream tasks that require dialogue comprehension abilities. Experiments show that our model outperforms strong pre-trained baselines, demonstrating that the learning-by-narrating strategy is a promising direction for dialogue comprehension. We also hope that DIANA will promote future research in related areas.

References

Stergos Afantenos, Eric Kow, Nicholas Asher, and Jérémy Perret. 2015. Discourse parsing for multi-party chat dialogues. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 928–937, Lisbon, Portugal. Association for Computational Linguistics.

Forough Arabshahi, Jennifer Lee, Mikayla Gawarecki, Kathryn Mazaitis, Amos Azaria, and Tom Mitchell. 2021. Conversational neuro-symbolic commonsense reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4902–4911.

Emile Chapuis, Pierre Colombo, Matteo Manica, Matthieu Labeau, and Chloé Clavel. 2020. Hierarchical pre-training for sequence labelling in spoken dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2636–2648, Online. Association for Computational Linguistics.

Jifan Chen, Eunsol Choi, and Greg Durrett. 2021. Can NLI models verify QA systems' predictions? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3841–3854, Punta Cana, Dominican Republic.

Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. 2020. MuTual: A dataset for multi-turn dialogue reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1406–1416, Online. Association for Computational Linguistics.

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

Tobias Falke, Markus Boese, Daniil Sorokin, Caglar Tirkaz, and Patrick Lehnen. 2020. Leveraging user paraphrasing behavior in dialog systems to automatically collect annotations for long-tail utterances. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pages 21–32, Online. International Committee on Computational Linguistics.

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

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719, New Orleans, Louisiana. Association for Computational Linguistics.

Qingqiu Huang, Yu Xiong, Anyi Rao, Jiaze Wang, and Dahua Lin. 2020. Movienet: A holistic dataset for movie understanding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 709–727. Springer.

Di Jin, Shuyang Gao, Jiun-Yu Kao, Tagyoung Chung, and Dilek Hakkani-Tür. 2020. MMM: multi-stage multi-task learning for multi-choice reading comprehension. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8010–8017. AAAI Press.

Jie Lei, Licheng Yu, Tamara Berg, and Mohit Bansal. 2020. What is more likely to happen next? video-and-language future event prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8769–8784, Online. Association for Computational Linguistics.

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

Pierre Lison, Jörg Tiedemann, and Milen Kouylekov. 2018. OpenSubtitles2018: Statistical rescoring of sentence alignments in large, noisy parallel corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Yongkang Liu, Shi Feng, Daling Wang, Kaisong Song, Feiliang Ren, and Yifei Zhang. 2021. A graph reasoning network for multi-turn response selection via customized pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13433–13442.

Kaixin Ma, Tomasz Jurczyk, and Jinho D. Choi. 2018. Challenging reading comprehension on daily conversation: Passage completion on multiparty dialog. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2039–2048, New Orleans, Louisiana. Association for Computational Linguistics.

Yao Qiu, Jinchao Zhang, and Jie Zhou. 2021. Different strokes for different folks: Investigating appropriate further pre-training approaches for diverse dialogue tasks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2318–2327, Online and Punta Cana, Dominican Republic.

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

Revanth Rameshkumar and Peter Bailey. 2020. Storytelling with dialogue: A Critical Role Dungeons and Dragons Dataset. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5121–5134, Online. Association for Computational Linguistics.

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

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, Yejin Choi, and Claire Cardie. 2019. DREAM: A challenge data set and models for dialogue-based reading comprehension. *Transactions of the Association for Computational Linguistics*, 7:217–231.

Makarand Tapaswi, Martin Bäuml, and Rainer Stiefelhagen. 2015. Aligning plot synopses to videos for story-based retrieval. *International Journal of Multimedia Information Retrieval*, 4(1):3–16.

Chien-Sheng Wu, Steven C.H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: Pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929, Online. Association for Computational Linguistics.

Wenpeng Yin, Dragomir Radev, and Caiming Xiong. 2021. DocNLI: A large-scale dataset for document-level natural language inference. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4913–4922, Online. Association for Computational Linguistics.

Leilan Zhang and Qiang Zhou. 2019. Automatically annotate tv series subtitles for dialogue corpus construction. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1029–1035. IEEE.

Zhuosheng Zhang and Hai Zhao. 2021. Structural pre-training for dialogue comprehension. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5134–5145, Online. Association for Computational Linguistics.

Kronecker Decomposition for GPT Compression

Ali Edalati
McGill University
ali.edalati@mail.mcgill.ca

Marzieh Tahaei
Huawei Noah Ark Lab
marzieh.tahaei@huawei.com

Ahmad Rashid
Huawei Noah Ark Lab
ahmad.rashid@huawei.com

Vahid Partovi Nia
Huawei Noah Ark Lab
vahid.partovinia@huawei.com

James J. Clark
McGill University
james.clark1@mcgill.ca

Mehdi Rezagholizadeh
Huawei Noah Ark Lab
mehdi.rezagholizadeh@huawei.com

Abstract

GPT is an auto-regressive Transformer-based pre-trained language model which has attracted a lot of attention in the natural language processing (NLP) domain. The success of GPT is mostly attributed to its pre-training on huge amount of data and its large number of parameters. Despite the superior performance of GPT, this overparameterized nature of GPT can be very prohibitive for deploying this model on devices with limited computational power or memory. This problem can be mitigated using model compression techniques; however, compressing GPT models has not been investigated much in the literature. In this work, we use Kronecker decomposition to compress the linear mappings of the GPT-2 model. Our Kronecker GPT-2 model (KnGPT2) is initialized based on the Kronecker decomposed version of the GPT-2 model and then is undergone a very light pre-training on only a small portion of the training data with intermediate layer knowledge distillation (ILKD). Finally, our KnGPT2 is fine-tuned on downstream tasks using ILKD as well. We evaluate our model on both language modeling and General Language Understanding Evaluation benchmark tasks and show that with more efficient pre-training and similar number of parameters, our KnGPT2 outperforms the existing DistilGPT2 model significantly.

1 Introduction

Recently, development and deployment of pre-trained language models (PLMs) has improved the performance of NLP models significantly (Devlin et al., 2018; Radford et al., 2019; Yang et al., 2019; Shoeybi et al., 2019; Radford et al., 2019). PLMs are mostly Transformer-based models, which are pre-trained on enormous unlabeled data. Although Transformer-based PLMs are powerful in performance, their huge size is a barrier for efficient training or inference of these models on lower capacity devices with memory, computation and energy constraints. Therefore, there has been a growing volume of literature focused on developing frameworks for compressing these large PLMs.

Like other deep learning models, the main directions of model compression for PLMs are using following methods in isolation or combination: low-bit quantization (Gong et al., 2014; Prato et al., 2019), pruning (Han et al., 2015), knowledge distillation (KD) (Hinton et al., 2015) and matrix decomposition (Yu et al., 2017; Lioutas et al., 2020).

PLMs can be divided into encoder-based and auto-regressive models such as the BERT (Devlin et al., 2018; Liu et al., 2019) and GPT (Brown et al., 2020) family respectively. Although the size of BERT family models is usually smaller than the GPT family, compressing the BERT family has been investigated much more in the literature (e.g. DistilBERT (Sanh et al., 2019), TinyBERT (Jiao et al., 2019), MobileBERT (Sun et al., 2020), ALP-KD (Passban et al., 2021), MATE-KD (Rashid et al., 2021), Annealing-KD (Jafari et al., 2021) and BERTQuant (Zhang et al., 2020)). On the other hand, to the best of our knowledge, the GPT family has barely a handful of compressed models (Li et al., 2021), among them the DistilGPT2[1] model has attracted wide attention in the literature. The DistilGPT2 model is heavily pre-trained for 3 epochs on the large OpenWebText dataset[2]. Moreover, it is evident in the literature that the GPT model cannot compete with BERT on natural language understanding (NLU) tasks (Liu et al., 2021). Therefore, developing an efficient compressed GPT model with comparable NLU performance is still an open problem.

In this paper, we use Kronecker decomposition, which has been recently used for BERT compression (Tahaei et al., 2021), for compression of the GPT-2 model (we refer to our model as KnGPT2 in this paper). We use Kronecker decomposition to represent the weight matrices of linear layers in

[1] https://transformer.huggingface.co/model/distil-gpt2

[2] https://huggingface.co/datasets/openwebtext

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 219 - 226
May 22-27, 2022 ©2022 Association for Computational Linguistics

GPT-2 by smaller matrices which can reduce the size and computation overhead. We use Kronecker decomposition to compress the embedding and Transformer layers of GPT-2. For Transformer layers, the linear layers of multi-head attention (MHA) and the feed-forward network (FFN) blocks of Transformer layers are replaced with their Kronecker decomposition.

Kronecker decomposition leads to reduction in expressiveness of the model. We use a very light pre-training with intermediate layer knowledge distillation (ILKD) to address this issue, which improves the performance of the compressed model significantly. It is worth mentioning that for our pre-training, we use $1/10^{\text{th}}$ of the DistilGPT2's pre-training data (i.e. OpenWebText) only for 1 epoch (instead of 3 epochs in DistilGPT2). Furthermore, in this paper, our framework is applied to GPT-2 but it can be easily exploited to compress other models as well. To summarize contributions of this paper, we mention the following points:

- To the best of our knowledge, we are the first work which uses Kronecker decomposition for compression of the GPT model.

- Our KnGPT2 model, which is evaluated on both language modeling and GLUE benchmark tasks, improves training efficiency and outperforms DistilGPT2 significantly.

2 Related Works

(Zhou and Wu, 2015) is the first work that used summation of multiple Kronecker products to compress the weight matrices in fully-connected networks and small convolutional neural networks. (Thakker et al., 2019) proposed a hybrid method which separates the weight matrices into an upper and a lower part, upper part remains untouched but the lower part decomposes to Kronecker products. They used this approach for small language models to be utilized on internet of things (IoT) applications. Recently, (Thakker et al., 2020) extended the mentioned hybrid method to non-IoT applications by adding a sparse matrix to the Kronecker products. (Tahaei et al., 2021) has deployed a similar approach to ours to compress BERT which achieved promising results but to the best of our knowledge, this work is the first attempt for GPT compression using Kronecker decomposition.

DistilGPT2 [3] is one of the most successful and well-known compressed versions of GPT-2 which is considered as a baseline in this paper. DistilGPT2 has 82M parameters compared to 124M parameters for GPT-2_{Small} and is trained using KD on OpenWebTextCorpus which is a reproduction of OpenAI's WebText dataset.

3 Methodology

3.1 Kronecker Product

The Kronecker product is a matrix operation (denoted by $\otimes$) which takes two matrices as input and generates a block matrix as output. Assume that $\mathbf{A}$ is a matrix $\in \mathbf{R}^{m_1 \times n_1}$ and $\mathbf{B}$ is a matrix $\in \mathbf{R}^{m_2 \times n_2}$, $\mathbf{A} \otimes \mathbf{B}$ is equal to a block matrix $\in \mathbf{R}^{m \times n}$, where $m = m_1 m_2$, $n = n_1 n_2$ and each block (i, j) is obtained by multiplying element a_{ij} by matrix .

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \cdots & a_{mn}\mathbf{B}, \end{bmatrix} \tag{1}$$

3.2 GPT-2 Compression using Kronecker Factorization

We can represent a weight matrix, $\mathbf{W} \in \mathbf{R}^{m \times n}$, by two smaller matrices, $\mathbf{A} \in \mathbf{R}^{m_1 \times n_1}$ and $\mathbf{B} \in \mathbf{R}^{m_2 \times n_2}$ such that $\mathbf{W} = \mathbf{A} \otimes \mathbf{B}$ and $m = m_1 m_2$, $n = n_1 n_2$. This leads to reduction in the number of parameters from mn for the original matrix to $m_1 n_1 + m_2 n_2$ for the Kronecker factorized version. In large language models, embedding layer usually takes a large portion of the memory. Let $\mathbf{W}^E \in \mathbf{R}^{v \times d}$ be the lookup table for the input embedding where v is the vocabulary size and d is the embedding dimension. To compress the embedding layer using Kronecker decomposition we use the same method as in (Tahaei et al., 2021). We define $\mathbf{A}^E \in \mathbf{R}^{v \times d/f}$ and $\mathbf{B}^E \in \mathbf{R}^{1 \times f}$, where f is a factor of d. There are two reasons for this decision: first, similar to $\mathbf{W}^E$, in the $\mathbf{A}^E$ matrix every row will indicate embedding of a single word. Second, the embedding of each word, E_i, can be obtained by $\mathbf{A}_i^E \otimes \mathbf{B}$, therefore the computation complexity of this operation is $\mathcal{O}(d)$ which is very efficient.

The transformer architecture is composed of N identical layers each having MHA followed by FFN. In the MHA module, there are linear layers which calculate the Query, Key and Value by

[3] For further details, see https://huggingface.co/distilgpt2

multiplying the input vector by $\mathbf{W}^{Q},^{K},^{V}$, respectively. Also, in the FFN module, there are two fully connected layers that can be represented as $\mathbf{W}^{c_{\text{fc}}}$ and $\mathbf{W}^{c_{\text{proj}}}$. In this work, all of the mentioned weight matrices at different heads and layers of the transformer are decomposed into Kronecker factors.

For initialization, similar to (Tahaei et al., 2021), the Kronecker factors $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ are estimated from the corresponding weight matrix $\mathbf{W}$ in the original uncompressed pre-trained model using the solution to the nearest Kronecker problem

$$(\hat{\mathbf{A}}, \hat{\mathbf{B}}) = \arg \min_{(\mathbf{A},\mathbf{B})} \|\mathbf{W} - \mathbf{A} \otimes \mathbf{B}\|^2$$

The solution to this optimization can be found by rank-1 singular value decomposition (SVD) approximation of the reshaped , see (Van Loan, 2000) for details.

3.3 Knowledge Distillation

In this section, the KD method used for both pre-training and fine-tuning the KnGPT2 is explained.

Let T and S represent the teacher model, GPT-2, and the student model, KnGPT2, respectively. For a batch of data $(\mathbf{x}, \mathbf{y})$, Att^{S}_{last} and Att^{T}_{last} are the attention distributions of the last transformer layer, obtained by applying softmax on the scaled dot product between query and key (For more details, see (Wang et al., 2020). H^{S}_{l} and H^{T}_{l} are the normalized hidden state outputs of the layer l. In our experiments, the output of the embedding layer is considered as a hidden state. Therefore, l represents both transformer layers and embedding layer. Note that by using the Kronecker factorization, like other decomposition methods, the number of layers and dimensions of the output matrices in the student model remain intact so we can directly obtain the difference of output of a specific layer in student an teacher model without the need for projection.

For the MHA modules, similar to (Wang et al., 2020), we use Kullback–Leibler divergence (KL) between the attention distributions of the last transformer layers of the student and the teacher.

$$L_{\text{Attention}}(x) = \text{KL}\{Att^{S}_{last}(x), Att^{T}_{last}(x)\} \quad (2)$$

For the FFN modules and embedding layer, we simply use the MSE between the output hidden states of embedding and transformer layers in the student and teacher:

$$L_{\text{Hidden States}}(x) = \frac{1}{L}\sum_{l} \text{MSE}\{H^{S}_{l}(x), H^{T}_{l}(x)\} \quad (3)$$

Where L is number of hidden states (number of transformer layers plus one for embedding).

The final loss is calculated by a linear combination of the above losses as well as the cross entropy loss.

$$\text{Loss}(x, y) = \sum_{(x,y)} \alpha_1 L_{\text{Attention}}(x) + \alpha_2 L_{\text{Hidden States}}(x) + \alpha_3 L_{\text{Cross Entropy}}(x, y) \quad (4)$$

After decomposing the teacher model, GPT-2, into KnGPT2, the performance of the model drops significantly. This drop is mainly because of the approximation of linear weight matrices using the corresponding Kronecker factors. Therefore, pre-training of the compressed model on a small corpus for a few epochs is necessary to retrieve the information which are lost during decomposition. Inspired by (Jiao et al., 2019), we pre-trained the model on a small portion, 10%, of the OpenWebText dataset (Gokaslan and Cohen, 2019) for one epoch and we used the KD method which is discussed in Section 3.3 to improve the performance of the compressed model.

4 Experiments

We evaluated our proposed algorithm, KnGPT2, on language modeling and text classification. For language modeling we use the Wikitext-103 (Merity et al.) dataset.For classification we use seven of the classification tasks of the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019). These datasets can be broadly divided into 3 families of problems. Single set tasks which include linguistic acceptability (CoLA) and sentiment analysis (SST-2), similarity and paraphrasing tasks (MRPC and QQP), and inference tasks which include Natural Language Inference (MNLI and RTE) and Question Answering (QNLI).

4.1 Experimental Setup

The KnGPT2 model is compressed from the GPT-2$_{\text{Small}}$ (Radford et al., 2019) model. GPT-2$_{\text{Small}}$ has 124 million parameters. Our baseline is DistilGPT2 which has about 82 million parameters so our KnGPT2 model is compressed to the same size (83 million parameters) for a fair

	GPT-2$_{\text{Small}}$	DistilGPT2	KnGPT2
Parameters*	124	82	83
Training time (hrs)	-	>90[4]	6.5
Dataset size (GB)	40	38	3.2

Table 1: Training details for GPT-2 compression. Note that number of parameters of the models are reported excluding the output embedding layer in language modelling which is not compressed, is equal to row Parameters*

Model	CoLA	RTE	MRPC	SST-2	MNLI	QNLI	QQP	Average
GPT-2$_{\text{Small}}$	44.0	63.2	84.5	92.8	81.75	88.7	88.0	77.56
DistilGPT2	32.4	61.9	84.3	90.8	79.55	85.4	87.3	74.52
DistilGPT2 + KD	33	61.5	84.4	90.7	79.85	85.7	87.6	74.67
KnGPT2	36.7	**64.4**	84.5	89.0	78.45	85.6	86.5	75.02
KnGPT2 + ILKD	**41.8**	63.7	**86.5**	**91.5**	**81.6**	**88.4**	**88.5**	**77.42**

Table 2: This table shows performance of the models on test set of GLUE tasks. Note that GPT-2$_{\text{Small}}$ is used as teacher for KD.

comparison. To achieve this, we compress half the layers of transformer block (odd numbered ones) in addition to the embedding layer by a factor of 2.

4.2 Pre-training

After the base model is compressed, performance of the compressed model drops significantly since the weight matrices with the Kronecker factors are approximate. Pre-training on a relatively small dataset for one epoch helps in retrieving the accuracy. Therefore, KnGPT2 is pre-trained (using the ILKD method discussed in Section 3.3) on 10% of OpenWebText which is 10 times less the DistiGPT2 model. As shown on Table 1 the training time for KnGPT2 is much faster as well.

	GPT-2$_{\text{Small}}$	DistilGPT2	KnGPT2
PPL	18.8	23.7	20.5

Table 3: Test Perplexity on WikiText-103.

4.3 Results

First we evaluate the models on language modeling using the Wikitext-103 dataset. The results are shown on Table 3. Although the DistilGPT2 is pre-trained longer and on a larger dataset the KnGPT2 achieves a lower perplexity.

Next, performance of the models is evaluated on the test (Table 2) sets of seven datasets of the GLUE benchmark. Similar to the pre-training, we used the ILKD method discussed in Section 3.3 to fine-tune KnGPT2. For DistilGPT, we apply the basic KD algorithm also referred to in the literature as Vanilla KD (Jafari et al., 2021). For DistilGPT since the number of layers between the teacher and the student are different, it is not clear which teacher layer should be distilled to which student layer. Although there has been work on intermediate distillation for mismatched layers for BERT (Passban et al., 2021), extensive experimentation is required to conclude the best practice for GPT.

On the test set results (Table 2), we observe that KnGPT2 outperforms DistilGPT2 for all datasets. Applying ILKD, even improves performance of KnGPT2. Another interesting result is that Vanilla KD does not improve DistilGPT2 fine-tuning. Interestingly KnGPT2 with KD reaches close to the GPT-2$_{\text{Small}}$ performance on average.

5 Conclusion

In this paper, we compressed GPT-2 by compressing linear layers of a GPT model using Kronecker decomposition. Our model is pre-trained on a relatively small (10 times smaller than the dataset used for baseline) dataset which makes the pre-training much faster. Our proposed model significantly outperformed the baseline on the GLUE benchmark. Using KD can help to further reduce the perfor-

[4] This number is presented in (Sanh et al., 2019) for training DistilBERT by the same authors. That uses the same KD algorithm and dataset for pre-training but is applied to BERT rather than GPT. Using a similar hardware we expect this number to be larger for DistilGPT

mance drop of the compressed model. Using Kronecker decomposition on larger GPT models and for higher compression factors are two interesting future directions.

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus.

Harvey Goldstein. 2011. *Multilevel statistical models*, volume 922. John Wiley & Sons.

Yunchao Gong, Liu Liu, Ming Yang, and Lubomir Bourdev. 2014. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*.

Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

Harold V Henderson, Friedrich Pukelsheim, and Shayle R Searle. 1983. On the history of the kronecker product. *Linear and Multilinear Algebra*, 14(2):113–120.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma, and Ali Ghodsi. 2021. Annealing knowledge distillation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2493–2504, Online. Association for Computational Linguistics.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. 2010. Kronecker graphs: an approach to modeling networks. *Journal of Machine Learning Research*, 11(2).

Tianda Li, Yassir El Mesbahi, Ivan Kobyzev, Ahmad Rashid, Atif Mahmud, Nithin Anchuri, Habib Hajimolahoseini, Yang Liu, and Mehdi Rezagholizadeh. 2021. A short study on compressing decoder-based language models. *arXiv preprint arXiv:2110.08460*.

Vasileios Lioutas, Ahmad Rashid, Krtin Kumar, Md Akmal Haidar, and Mehdi Rezagholizadeh. 2020. Improving word embedding factorization for compression using distilled nonlinear neural decomposition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2774–2784.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2021. ALP-KD: attention-based layer projection for knowledge distillation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13657–13665. AAAI Press.

Gabriele Prato, Ella Charlaix, and Mehdi Rezagholizadeh. 2019. Fully quantized transformer for machine translation. *arXiv preprint arXiv:1910.10485*.

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

Ahmad Rashid, Vasileios Lioutas, and Mehdi Rezagholizadeh. 2021. Mate-kd: Masked adversarial text, a companion to knowledge distillation. *arXiv preprint arXiv:2105.05912*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

Marzieh S. Tahaei, Ella Charlaix, Vahid Partovi Nia, Ali Ghodsi, and Mehdi Rezagholizadeh. 2021. Kroneckerbert: Learning kronecker decomposition for pre-trained language models via knowledge distillation.

Urmish Thakker, Jesse Beu, Dibakar Gope, Chu Zhou, Igor Fedorov, Ganesh Dasika, and Matthew Mattina. 2019. Compressing rnns for iot devices by 15-38x using kronecker products. *arXiv preprint arXiv:1906.02876*.

Urmish Thakker, Paul Whatamough, Matthew Mattina, and Jesse Beu. 2020. Compressing language models using doped kronecker products. *arXiv preprint arXiv:2001.08896*.

Charles F Van Loan. 2000. The ubiquitous kronecker product. *Journal of computational and applied mathematics*, 123(1-2):85–100.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *arXiv preprint arXiv:2002.10957*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. 2017. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7370–7379.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020. Ternarybert: Distillation-aware ultra-low bit bert. *arXiv preprint arXiv:2009.12812*.

Shuchang Zhou and Jia-Nan Wu. 2015. Compression of fully-connected layer in neural network by kronecker product. *arXiv preprint arXiv:1507.05775*.

A Configurations

Table 4 shows sizes of matrices in GPT-2_{Small}, DistilGPT2 and KnGPT2.

B Kronecker product further explanation

Kronecker product has attractive abstract algebraic properties such as

$$\mathbf{A} \otimes (\mathbf{B} + \mathbf{C}) = \mathbf{A} \otimes \mathbf{B} + \mathbf{A} \otimes \mathbf{C}$$

$$(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}$$

$$(\mathbf{A} \otimes \mathbf{B})^{\top} = \mathbf{A}^{\top} \otimes \mathbf{B}^{\top}$$

for more details see (Henderson et al., 1983). The interesting properties of the Kronecker product makes it an attractive tool for decomposition of large matrices. The Kronecker product is also a flexible method to simplify the notation of large block matrices, both in linear mixed effect models and multilevel models (Goldstein, 2011). It is also a well-known technique to represent large repetitive structured graphs using the Kronecker product (Leskovec et al., 2010). One of the most important characteristics of a matrix is its determinant and it is well-known that for two square matrices $\mathbf{A}$ and $\mathbf{B}$ of size n, and m, $|\mathbf{A} \otimes \mathbf{B}| = |\mathbf{A}|^n |\mathbf{B}|^m$. This property explains the superiority of Kronecker compared to the other decomposition methods for large matrices. By choosing the right n and m, a large matrix $\mathbf{W} = \mathbf{A} \otimes \mathbf{B}$ can be decomposed to much smaller matrices such that the above determinant equation holds.

C Hyperparameters

Table 5 shows tuned hyperparameters for pre-training on OpenWebText and fine-tuning on GLUE. Also, to fine-tune models on Wikitext-103, we used the same hyperparamters as pre-training.

D GLUE dev results

Table 6 show performance of the models on dev set of GLUE.

E Ablation Study

(Tahaei et al., 2021) uses MSE as the distance metric between output of attention layers from all of the transformer layers of teacher and student. One of differences of our work from (Tahaei et al., 2021) is that inspired from (Wang et al., 2020), we used Kullback–Leibler (KL) divergence of the output of last attention layer. Using KL divergence over MSE improved average performance of the model, for 0.92% on the following GLUE tasks: COLA, MNLI, MRPC, QNLI, QQP, RTE and SST2 (Table 7).

(Tahaei et al., 2021) only minimizes the KD at the pre-training stage. we performed two experiments to study the effect of KD on the pre-training of KnGPT2 to improve performance of our model. In the first experiment, KnGPT2 is pre-trained by KD loss, with and without cross entropy (CE) loss

Model	Embedding	Q,K,V	FFN*
GPT-2$_{\text{Small}}$	50527×768	768×768	3072×768
DistilGPT2	50527×768	768×768	3072×768
KnGPT2	$A : 50527 \times 384$, $B : 1 \times 2$	$A : 384 \times 768$, B:2×1	$A : 1536 \times 768$, $B : 2 \times 1$

Table 4: This table shows configuration of the models. Note that FFN block has two projections that shape of one is the transpose of the other one and here, only shape of one of them is mentioned. Also, for KnGPT2, mentioned shapes for transformer layer belong to half of the layers that are decomposed -layers with odd numbers- and shape of the other half are the same with the GPT-2 model.

Phase	Epoch	Sequence Length	Seed	Batch size	Learning rate	α_1	α_2	α_3
Pre-training	1	1024	42	1	0.00025	0.5	0.5	0.1
Fine-tuning	20	128	42	16	2e-5	0.5	0.5	0.02

Table 5: hyper-parameters that are used for pre-training and fine-tuning.

Model	CoLA	RTE	MRPC	SST-2	MNLI	QNLI	QQP	Average
GPT-2$_{\text{Small}}$	47.6	69.31	87.47	92.08	83.12	88.87	90.25	79.81
DistilGPT2	38.7	65.0	87.7	91.3	79.9	85.7	89.3	76.8
DistilGPT2 + KD	38.64	64.98	87.31	89.80	80.42	86.36	89.61	76.73
KnGPT2	37.51	**70.4**	**88.55**	88.64	78.93	86.10	88.87	77
KnGPT2 + ILKD	**45.36**	69.67	87.41	**91.28**	**82.15**	**88.58**	**90.34**	**79.25**

Table 6: This table shows performance of the models on dev set of GLUE tasks. Note that GPT-2$_{\text{Small}}$ is used as teacher for KD.

Model	CoLA	RTE	MRPC	SST-2	MNLI	QNLI	QQP	Average
KnGPT2 + ILKD$_{MSE}$	41.65	68.95	**88.89**	90.48	80.69	87.66	90.00	78.33
KnGPT2 + ILKD$_{KL}$	**45.36**	**69.67**	87.41	**91.28**	**82.15**	**88.58**	**90.34**	**79.25**

Table 7: This table shows performance of the Kronecker models (on dev set of GLUE tasks) that are fine-tuned using MSE and KL divergence as the distance metric in Equation 2.

Model	α_3	CoLA	RTE	MRPC	SST-2	MNLI	QNLI	QQP	Average
KnGPT2 + ILKD	0	40.80	**70.04**	**88.25**	90.71	80.12	87.64	89.64	78.17
KnGPT2 + ILKD	0.1	**45.36**	**69.67**	87.41	**91.28**	**82.15**	**88.58**	**90.34**	**79.25**

Table 8: This table shows performance of the Kronecker models (on dev set of GLUE tasks) that are pre-trianed with and without CE loss. α_3 indicates coefficient of CE loss during pre-training.

Model	Wikitext-103(PPL)	MNLI (f1)
KnGPT2	28608	69.33
KnGPT2 + LM	**21.94**	77.87
KnGPT2 + KD	144.1	77.50
KnGPT2 + LM + KD	23.04	**77.97**

Table 9: Ablation on the effect of pre-training with KD on language model and MNLI classification

then fine-tuned on GLUE with mentioned ILKD method discussed in Section 3.3. Empirical results (Table 8) show that adding cross entropy loss improves performance of Kronecker model on downstream tasks so we used KD + CE loss for both pre-training and fine-tuning. In the second experiment we used Wikitext-103 as our pre-training dataset. We compare four models and evaluate on LM as well as on classification using the MNLI dataset from GLUE. As shown on Table 9 we compare KnGPT2 without pre-training, with language modeling pre-training only, with KD pre-training only and both language modeling and KD pre-training. Note that we apply ILKD, discussed before for fine-tuning, as our KD algorithm. We observe that pre-training is important for good performance on the downstream task but lower perplexity on LM is not always a good indicator of better downstream performance.

Simple and Effective Knowledge-Driven Query Expansion for QA-Based Product Attribute Extraction

Keiji Shinzato
Rakuten Institute of Technology,
Rakuten Group Inc.
keiji.shinzato@rakuten.com

Naoki Yoshinaga
Institute of Industrial Science,
the University of Tokyo
ynaga@iis.u-tokyo.ac.jp

Yandi Xia
Rakuten Institute of Technology,
Rakuten Group Inc.
yandi.xia@rakuten.com

Wei-Te Chen
Rakuten Institute of Technology,
Rakuten Group Inc.
weite.chen@rakuten.com

Abstract

A key challenge in attribute value extraction (AVE) from e-commerce sites is how to handle a large number of attributes for diverse products. Although this challenge is partially addressed by a question answering (QA) approach which finds a value in product data for a given query (attribute), it does not work effectively for rare and ambiguous queries. We thus propose simple knowledge-driven query expansion based on possible answers (values) of a query (attribute) for QA-based AVE. We retrieve values of a query (attribute) from the training data to expand the query. We train a model with two tricks, knowledge dropout and knowledge token mixing, which mimic the imperfection of the value knowledge in testing. Experimental results on our cleaned version of AliExpress dataset show that our method improves the performance of AVE (+6.08 macro F_1), especially for rare and ambiguous attributes (+7.82 and +6.86 macro F_1, respectively).

1 Introduction

One of the most challenging problems in attribute value extraction (AVE) from e-commerce sites is a data sparseness problem caused by the diversity of attributes.[1] To alleviate the data sparseness problem, recent researches (Xu et al., 2019; Wang et al., 2020) formalize the task as question answering (QA) to exploit the similarity of attributes via representation learning. Specifically, the QA-based AVE takes an attribute name as *query* and product data as *context*, and attempts to extract the value from the context. Although this approach mitigates the data sparseness problem, performance depends on the quality of query representations (Li et al., 2020).

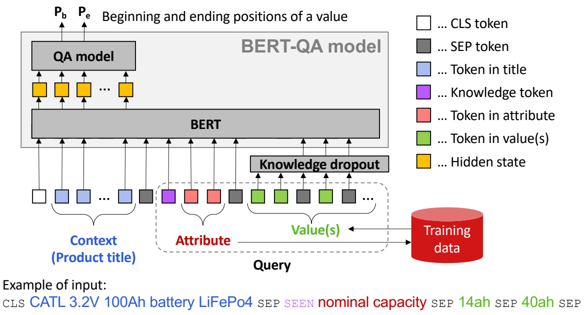


Figure 1: Our knowledge-based BERT-QA model for attribute value extraction.

Because attribute names are short and ambiguous as queries, the extraction performance drops significantly for rare attributes with ambiguous names (*e.g.*, *sort*) which do not represent their values well.

Aiming to perform more accurate QA-based AVE for rare and ambiguous attributes, we propose simple query expansion that exploits values for the attribute as knowledge to learn better query representations (Figure 1, § 3). We first retrieve possible values of each attribute from the training data, and then use the obtained values to augment the query (attribute). Since unseen values and attributes will appear in evaluation, we apply dropout to the seen values to mimic the incompleteness of the knowledge (§ 3.2), and perform multi-domain learning to capture the absence of the knowledge (§ 3.3).

We demonstrate the effectiveness of the query expansion for BERT-based AVE model (Wang et al., 2020) using the AliExpress dataset[2] released by Xu et al. (2019) (§ 4). In the evaluation process, we found near-duplicated data in this dataset. We thus construct, from this dataset, a more reliable dataset called cleaned AE-pub to evaluate our method.

[1] AliExpress.com classifies products in the Sports & Entertainment category using 77,699 attributes (Xu et al., 2019).

[2] https://github.com/lanmanok/ACL19_Scaling_Up_Open_Tagging

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 227 - 234
May 22-27, 2022 ©2022 Association for Computational Linguistics

Our contribution is threefold:

- We proposed knowledge-driven query expansion for QA-based AVE (§ 3); the knowledge taken from the training data is valuable (§ 4.3).
- We revealed that rare, ambiguous attributes deteriorate the performance of QA-based AVE in the e-commerce domain (§ 4.3).
- We will release our cleaned version of AliExpress dataset for research purposes.

2 Related Work

Attribute value extraction has been modeled as a sequence labeling problem (Putthividhya and Hu, 2011; Shinzato and Sekine, 2013; More, 2016; Zheng et al., 2018; Rezk et al., 2019; Karamanolakis et al., 2020; Dong et al., 2020; Zhu et al., 2020; Mehta et al., 2021; Jain et al., 2021; Yan et al., 2021). However, since the number of attributes can exceed ten thousand in e-commerce sites, the models perform poorly for the majority of attributes that rarely appear in the labeled data (Xu et al., 2019).

To alleviate the data sparseness problem, Xu et al. (2019) introduced a QA-based approach for the AVE task. It separately encodes product titles and attributes using BERT (Devlin et al., 2019) and bi-directional long-short term memory (Hochreiter and Schmidhuber, 1997), and then combines the resulting vectors via an attention layer to learn spans of values for the attributes from the titles. Wang et al. (2020) proposed a purely BERT-based model, which feeds a string concatenating the given title and attribute to BERT. These QA-based AVE models, however, do not fully enjoy the advantage of the QA model, since attribute queries are much shorter than sentential questions in the original QA task.

To build better queries in solving named entity recognition via QA, Li et al. (2020) exploited annotation guideline notes for named entity classes as queries. Although this approach will be also effective for QA-based AVE, it requires substantial labors to prepare manual annotations for more than ten thousand attributes in e-commerce site.

3 Proposed Method

This section proposes a simple but effective query expansion method for QA-based AVE (Wang et al., 2020) by utilizing attribute values. Given a product data (title) $\boldsymbol{x} = \{x_1, ..., x_n\}$ and an attribute $\boldsymbol{a} = \{a_1, ..., a_m\}$, where n and m denote the number of tokens, the model returns the beginning position, P_b, and ending position, P_e, of a value.

Figure 1 depicts the model architecture with our approach. Although our query expansion is essentially applicable to any QA-based AVE models, we here employ the state-of-the-art model using BERT proposed by Wang et al. (2020). In addition to the QA component for AVE, their model has other two components; the no-answer classifier and the distilled masked language model. Since those components slightly decrease the overall micro F_1, we employ the QA component from their model (hearafter, referred to as BERT-QA).

3.1 Knowledge-Driven Query Expansion for QA-Based AVE

It is inherently difficult for QA-based AVE models to induce effective query representations for rare attributes with ambiguous names. It is also hard to develop expensive resources such as annotation guideline notes (Li et al., 2020) for more than ten thousand of attributes in e-commerce domain.

Then, is there any low-cost resource (knowledge) we can leverage to understand attributes? Our answer to this question is values (answers) for the attributes; we can guess what attributes means from their values. In this study, we exploit attribute values retrieved from the training data[3] of the target AVE model as *run-time knowledge* to induce better query representations.

Our query expansion allows the QA-based AVE model, M_{QA}, to utilize the seen values for attribute $\boldsymbol{a}$ in the whole training data to find beginning and ending positions of a value, $\langle P_b, P_e\rangle$ in title $\boldsymbol{x}$:

$$\langle P_b, P_e\rangle = M_{\mathrm{QA}}([\mathrm{CLS}; \boldsymbol{x}; \mathrm{SEP}; \boldsymbol{a}; \mathrm{SEP}; \boldsymbol{v_a}]) \quad (1)$$

Here, CLS and SEP are special tokens to represent a classifier token and a separator, respectively, and $\boldsymbol{v_a}$ is a string concatenating the seen values of the attribute $\boldsymbol{a}$ with SEP in descending order of frequency in the training data.

3.2 Knowledge Dropout

By taking all the seen values in the training data to augment input queries, the model may just learn to match the seen values with one in the given title. To avoid this, inspired from word dropout employed in language modeling (Gal and Ghahramani, 2016),

[3] We can utilize, if any, external resources for our method. For example, e-commerce sites may develop attribute-value databases to organize products in the marketplace.

we perform *knowledge dropout* over $\boldsymbol{v_a}$ in training before concatenating it with title $\boldsymbol{x}$ and attribute $\boldsymbol{a}$.

$$\boldsymbol{v_a} = [\text{drop}(\boldsymbol{v_{a,1}}); \text{SEP}; \text{drop}(\boldsymbol{v_{a,2}}); \text{SEP}; \ldots] \quad (2)$$

Here, drop is a function that replaces a value $\boldsymbol{v_{a,i}}$ in $\boldsymbol{v_a}$ with padding tokens according to a dropout rate; we replace each token in $\boldsymbol{v_{a,i}}$ with PAD.

To decide if the dropout applies to a value, we take account of the number of examples labeled with the value. Given the dropout rate r and the number of training examples n_v, the dropout performs over the value $\boldsymbol{v}$ according to the probability of r^{n_v}. This implementation captures the fact that infrequent values are more likely to be unseen.

3.3 Knowledge Token Mixing

Since values are literally valuable to interpret attributes, the QA-based AVE model may rely more on values than an attribute name. This will hurt the performance on unseen attributes whose values are not available. To avoid this, we assume the availability of value knowledge to be *domain*, and perform multi-domain learning for QA-based model with and without our value-based query expansion. This will allow the model to handle not only seen attributes but also unseen attributes.

Inspired from domain token mixing (Britz et al., 2017), we introduce two special domain tokens (*knowledge tokens*), and prepend either of the tokens to the attribute to express the knowledge status: SEEN and UNSEEN (with and without values).[4] In training, from an example with title $\boldsymbol{x}$ and attribute $\boldsymbol{a}$, we build [CLS;$\boldsymbol{x}$;SEP;SEEN;$\boldsymbol{a}$;SEP;$\boldsymbol{v_a}$] and [CLS;$\boldsymbol{x}$;SEP;UNSEEN;$\boldsymbol{a}$;SEP], and then put these examples to the same mini-batch. In testing, we use SEEN and UNSEEN tokens for seen attributes (with values) and unseen attributes, respectively.

4 Experiments

We evaluate our query expansion method for QA-based AVE on a public dataset,[2] which is built from product data under the Sports & Entertainment category in AliExpress, following (Wang et al., 2020).

4.1 Settings

Dataset The public AliExpress dataset consists of 110,484 tuples of ⟨product title, attribute, value⟩. When a value of the attribute is absent from the title, the value in the tuple is set as "NULL." We manually inspected the tuples in the dataset, and found quality issues; some tuples contained HTML entities, and extra white spaces in titles, attributes, and values, and the same attributes sometimes have different letter cases. We thus decoded HTML entities, converted trailing spaces into a single space, and removed white spaces at the beginning and ending. We also normalized the attributes by putting a space between alphabets and numbers and by removing ':' at the endings (from 'feature1:' to 'feature 1'). As a result, we found 736 duplicated tuples. By removing these duplicated tuples, we finally obtained the *cleaned* AE-*pub* dataset of 109,748 tuples with 2,162 unique attributes and 11,955 unique values. We split this dataset into training, development, and test sets with the ratio of 7:1:2 (Table 1).

	Train	Dev.	Test
# of tuples	76,823	10,975	21,950
# of tuples with "NULL"	15,097	2,201	4,259
# of unique attribute-value pairs	11,819	2,680	4,431
# of unique attributes	1,801	635	872
# of unique values	9,317	2,258	3,671
# of tuples (Wang et al., 2020)	88,479	N/A	22,005

Table 1: Statistics of the cleaned AE-pub dataset.

Evaluation Metrics We use precision (P), recall (R) and F$_1$ score as metrics. We adopt exact match criteria (Xu et al., 2019) in which the full sequence of extracted value needs to be correct.

4.2 Models

We apply our knowledge-driven query expansion method (§ 3) to BERT-QA (Wang et al., 2020), a QA-based AVE model on BERT. To perform the query expansion, we simply collect values other than "NULL" from tuples in the training data for each attribute (Table 1).

For comparison, we use SUOpenTag (Xu et al., 2019), AVEQA and vanilla BERT-QA (Wang et al., 2020), which achieved the state-of-the-art micro F$_1$ score on the AliExpress dataset. We also perform a simple dictionary matching; it returns the most frequent seen value for a given attribute among those included in the given title.

To convert tuples in the training set to beginning and ending positions, we tokenize both title and value, and then use matching positions if the token sequence of the value exactly matches a subsequence of the title. If the value matches multiple

[4] The original domain token mixing learns to induce domain tokens prior to generating outputs, whereas we prepend domain tokens to inputs since the knowledge status is known.

Models	Macro P (%)	 R (%)	 F$_1$	Micro P (%)	 R (%)	 F$_1$
Dictionary	33.20 (±0.00)	30.37 (±0.00)	31.72 (±0.00)	73.39 (±0.00)	73.77 (±0.00)	73.58 (±0.00)
SUOpenTag (Xu et al., 2019)	30.92 (±1.44)	28.04 (±1.48)	29.41 (±1.44)	86.53 (±0.78)	79.11 (±0.35)	82.65 (±0.20)
AVEQA (Wang et al., 2020)	41.93 (±1.05)	39.65 (±0.96)	40.76 (±0.98)	86.95 (±0.27)	81.99 (±0.13)	84.40 (±0.09)
BERT-QA (Wang et al., 2020)	42.77 (±0.36)	40.85 (±0.22)	41.79 (±0.28)	87.14 (±0.54)	82.16 (±0.21)	84.58 (±0.24)
BERT-QA +vals	39.48 (±0.37)	35.60 (±0.44)	37.44 (±0.38)	**88.82** (±0.22)	81.77 (±0.14)	85.15 (±0.14)
BERT-QA +vals +drop	41.61 (±0.83)	38.22 (±0.80)	39.84 (±0.81)	88.46 (±0.26)	82.02 (±0.37)	85.12 (±0.14)
BERT-QA +vals +mixing	46.67 (±0.33)	43.32 (±0.50)	44.93 (±0.39)	88.30 (±0.69)	82.46 (±0.30)	**85.28** (±0.26)
BERT-QA +vals +drop +mixing	**47.74** (±0.54)	**44.82** (±0.75)	**46.23** (±0.64)	87.84 (±0.39)	**82.61** (±0.07)	85.14 (±0.19)

Table 2: Performance on the cleaned AE-pub dataset in Table 1; reported numbers are mean (std. dev.) of five trials.

Models	cos	Macro Number of training examples (median: 8) $[1, 8)$	 $[8, \infty)$	 all	Micro Number of training examples (median: 8) $[1, 8)$	 $[8, \infty)$	 all
BERT-QA +vals	lo	42.41 (+0.80)	57.58 (+5.84)	49.96 (+3.31)	55.65 (+8.34)	78.51 (+1.33)	77.59 (+1.77)
	hi	41.05 (−1.75)	69.02 (+1.00)	56.58 (−0.23)	57.72 (+6.04)	88.54 (−0.06)	88.21 (+0.05)
	all	41.77 (−0.40)	63.63 (+3.29)	53.28 (+1.55)	56.65 (+7.25)	86.39 (+0.25)	85.89 (+0.46)
BERT-QA +vals +drop	lo	45.34 (+3.73)	57.89 (+6.15)	51.58 (+4.93)	58.61 (+11.30)	78.70 (+1.52)	77.87 (+2.05)
	hi	45.21 (+2.41)	70.11 (+2.09)	59.04 (+2.23)	60.71 (+9.03)	88.45 (−0.15)	88.16 (±0.00)
	all	45.28 (+3.11)	64.35 (+4.01)	55.32 (+3.59)	59.62 (+10.22)	86.37 (+0.23)	85.91 (+0.48)
BERT-QA +vals +mixing	lo	47.64 (+6.03)	57.91 (+**6.17**)	52.75 (+6.10)	58.13 (+10.82)	78.78 (+**1.60**)	77.90 (+**2.08**)
	hi	48.38 (+5.58)	70.48 (+2.46)	60.67 (+3.86)	62.10 (+10.42)	88.74 (+**0.14**)	88.45 (+**0.29**)
	all	47.99 (+5.82)	64.55 (+4.21)	56.71 (+4.98)	60.03 (+10.63)	86.61 (+**0.47**)	86.13 (+**0.70**)
BERT-QA +vals +drop +mixing	lo	49.15 (+**7.54**)	57.89 (+6.15)	53.51 (+**6.86**)	60.18 (+**12.87**)	78.55 (+1.37)	77.74 (+1.92)
	hi	50.94 (+**8.14**)	71.04 (+**3.02**)	62.10 (+**5.29**)	63.06 (+**11.38**)	88.56 (−0.04)	88.27 (+0.11)
	all	49.99 (+**7.82**)	64.84 (+**4.50**)	57.81 (+**6.08**)	61.56 (+**12.16**)	86.42 (+0.28)	85.96 (+0.53)

Table 3: Macro and micro F$_1$ gains over BERT-QA for 544 attributes (21,374 test examples) that took our value-based query expansion. 'lo' and 'hi' are similarity intervals, $[0.411, 0.929)$ and $[0.929, 1.0]$, respectively.

portions of the title, we use the match close to the beginning of the title. As beginning and ending positions of tuples whose value is "NULL," we use 0 which is a position of a CLS token in the title. The conversion procedure is detailed in Appendix A.1.

We implemented the above models using PyTorch (Paszke et al., 2019) (ver. 1.7.1), and used "bert-base-uncased" in Transformers (Wolf et al., 2020) as the pre-trained BERT (BERT$_{\text{BASE}}$). The implementation details and the training time are given in Appendix A.2 and Appendix A.3, respectively.

4.3 Results

Table 2 shows macro[5] and micro performance of each model that are averaged over five trials. The low recall of the model BERT-QA +vals suggests that this model learns to find strings that are similar to ones retrieved from the training data (overfitting). On the other hand, knowledge dropout and knowledge token mixing mitigates the overfitting, and improves both macro and micro F$_1$ performance.

Impact on rare and ambiguous attributes To see if the query expansion improves the performance for rare attributes with ambiguous names, we categorized the attributes that took the query expansion according to the number of training examples and the appropriateness of the attribute names for their values. To measure the name appropriateness, we exploit embeddings of the CLS token using the BERT$_{\text{BASE}}$ for each attribute and its seen values; when the cosine similarity between the attribute embedding and averaged value embeddings is low, we regard the attribute name as ambiguous. We divide the attributes into four according to median frequency and similarity to values.

Table 3 lists macro and micro F$_1$ of each model and the improvements over the BERT-QA for each category. We can see that our query expansion tends to be more effective for attributes with low similarity. This means that the query expansion can generate more informative queries than ambiguous attributes alone. Moreover, by using knowledge dropout and knowledge token mixing, we can improve macro and micro F$_1$ for rare attributes. These

[5] We ignored 70 attributes with only NULL since we cannot compute recall and F$_1$ for these attributes.

Models	Seen Attr. (Seen Values)		Seen Attr. (Unseen Values)		Unseen Attr.	
	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1
Dictionary	87.19 (±0.00)	83.89 (±0.00)	n/a	n/a	n/a	n/a
BERT-QA (Wang et al., 2020)	92.26 (±0.15)	73.30 (±2.30)	46.11 (±0.86)	**25.43** (±1.24)	25.86 (±2.53)	20.92 (±1.92)
BERT-QA +vals	**92.85** (±0.13)	**86.93** (±0.61)	42.11 (±0.70)	10.98 (±1.01)	6.90 (±1.19)	4.03 (±0.76)
BERT-QA +vals +drop	92.74 (±0.20)	86.21 (±0.58)	44.14 (±0.19)	16.40 (±0.80)	11.17 (±2.89)	7.21 (±2.16)
BERT-QA +vals +mixing	92.82 (±0.15)	86.40 (±0.79)	45.59 (±0.48)	19.87 (±1.81)	25.39 (±2.63)	20.14 (±2.13)
BERT-QA +vals +drop +mixing	92.67 (±0.11)	86.34 (±0.72)	**46.14** (±0.34)	22.52 (±0.93)	**27.54** (±1.35)	**21.95** (±1.25)

Table 4: Performance on the cleaned AE-pub dataset in terms of the types of the attribute values; reported numbers are mean (std. dev.) of five trials. The best score is in bold face and the second best score is underlined.

results are remarkable since the knowledge used to enhance the model comes from its training data; the model could use more parameters to solve the task itself by taking the internal knowledge induced from the training data as runtime input.

Impact on seen and unseen attribute values To see for what types of attribute values the query expansion is effective, we categorize the test examples according to the types of the training data used to solve the examples. We first categorize the test examples into seen or unseen attributes. Next, we further classify the examples for the seen attributes into either seen or unseen attribute values.

Table 4 shows the performance in terms of the attribute value types. The query expansion improved macro F_1 by 13 points on the seen values for the seen attributes; these improvements were yielded by the large performance gains for rare attributes in Table 3. Although BERT-QA +vals performed the best on the seen values, it performed the worst on the unseen values for the seen attributes and unseen attributes; the model is trained to match seen values in a query with a given title. Meanwhile, the two tricks enable the model to maintain the micro F_1 performance of BERT on the unseen values for the seen attributes. The lower macro F_1 against BERT suggests that there is still room for improvements in query representation for rare seen attributes. Lastly, the knowledge token mixing successfully recovered the performance of BERT for the unseen attributes, and even improved the performance when it is used together with the knowledge dropout. This is possibly because the knowledge token mixing allows the model to switch its behavior for seen and unseen attributes, and the knowledge dropout strengthens the ability to induce better query representations.

Example outputs Table 5 shows examples of the actual model outputs for a given context and query (attribute (seen values)). In the first two examples, *function 1* and *nominal capacity* are ambiguous and rare attributes, respectively, and are thereby hard for the BERT-QA to extract **correct** values without the help of our query expansion. As shown in the last example, when there are more than one candidates as values of a given attribute, our query expansion is still unstable.

C: aeronova [bicycle [**carbon mtb handlebar**]$_{\text{ours}}$]$_{\text{BERT-QA}}$ mountain bikes flat handlebar mtb integrated handlebars with stem bike accessories
Q: function 1 (skiing goggles, carbon road bicycle handlebar, cycling glasses, bicycle mask, gas mask, . . .)

C: lfp [3.2v [**100ah**]$_{\text{ours}}$]$_{\text{BERT-QA}}$ lifepo4 prismatic cell deep cycle diy lithium ion battery 72v 60v 48v 24v 100ah 200ah ev solar storage battery
Q: nominal capacity (14ah, 40ah, 17.4ah)

C: camel outdoor softshell [**men**]$_{\text{BERT-QA}}$'s hiking jacket windproof thermal jacket for [camping]$_{\text{ours}}$ ski thick warm coats
Q: suitable (men, camping, kids, saltwater/freshwater, women, 4-15y, mtb cycling shoes, . . .)

Table 5: Example outputs of BERT-QA with and without query expansion for given C(ontext) and Q(uery).

5 Conclusions

We have proposed simple query expansion based on possible values of a given query (attribute) for QA-based attribute extraction. With the two tricks to mimic the imperfection of the value knowledge, we retrieve values of given attributes from the training data, and then use the obtained values as knowledge to induce better query representations. Experimental results on our cleaned version of the public AliExpress dataset demonstrate that our method improves the performance of product attribute extraction, especially for rare and ambiguous attributes.

We will leverage external resources to handle unseen attributes (preliminary experiments are shown in Appendix A.4). We will release the script to build our cleaned AE-pub dataset.[6]

[6] http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/acl2022/

Acknowledgements

This work (second author) was partially supported by JSPS KAKENHI Grant Number 21H03494. We thank the anonymous reviewers for their hard work.

References

Denny Britz, Quoc Le, and Reid Pryzant. 2017. Effective domain mixing for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 118–126, Copenhagen, Denmark. Association for Computational Linguistics.

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

Xin Luna Dong, Xiang He, Andrey Kan, Xian Li, Yan Liang, Jun Ma, Yifan Ethan Xu, Chenwei Zhang, Tong Zhao, Gabriel Blanco Saldana, Saurabh Deshpande, Alexandre Michetti Manduca, Jay Ren, Surender Pal Singh, Fan Xiao, Haw-Shiuan Chang, Giannis Karamanolakis, Yuning Mao, Yaqing Wang, Christos Faloutsos, Andrew McCallum, and Jiawei Han. 2020. Autoknow: Self-driving knowledge collection for products of thousands of types. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 2724–2734, New York, NY, USA. Association for Computing Machinery.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems 29 (NIPS)*.

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

Mayank Jain, Sourangshu Bhattacharya, Harshit Jain, Karimulla Shaik, and Muthusamy Chelliah. 2021. Learning cross-task attribute - attribute similarity for multi-task attribute-value extraction. In *Proceedings of The 4th Workshop on e-Commerce and NLP*, pages 79–87, Online. Association for Computational Linguistics.

Giannis Karamanolakis, Jun Ma, and Xin Luna Dong. 2020. TXtract: Taxonomy-aware knowledge extraction for thousands of product categories. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8489–8502, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the third International Conference on Learning Representations*, San Diego, California, USA.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Kartik Mehta, Ioana Oprea, and Nikhil Rasiwasia. 2021. LATEX-numeric: Language agnostic text attribute extraction for numeric attributes. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, pages 272–279, Online. Association for Computational Linguistics.

Ajinkya More. 2016. Attribute extraction from product titles in ecommerce. In *KDD 2016 Workshop on Enterprise Intelligence*, San Francisco, California, USA.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., Red Hook, NY, USA.

Duangmanee Putthividhya and Junling Hu. 2011. Bootstrapped named entity recognition for product attribute extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Martin Rezk, Laura Alonso Alemany, Lasguido Nio, and Ted Zhang. 2019. Accurate product attribute extraction on the field. In *Proceedings of the 35th IEEE International Conference on Data Engineering*, pages 1862–1873, Macau SAR, China. IEEE.

Keiji Shinzato and Satoshi Sekine. 2013. Unsupervised extraction of attributes and their values from product description. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1339–1347, Nagoya, Japan. Asian Federation of Natural Language Processing.

Qifan Wang, Li Yang, Bhargav Kanagal, Sumit Sanghai, D. Sivakumar, Bin Shu, Zac Yu, and Jon Elsas. 2020. Learning to extract attribute value from product via question answering: A multi-task approach. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 47–55, Online. ACM.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 38–45, Online. Association for Computational Linguistics.

Huimin Xu, Wenting Wang, Xin Mao, Xinyu Jiang, and Man Lan. 2019. Scaling up open tagging from tens to thousands: Comprehension empowered attribute value extraction from product title. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5214–5223, Florence, Italy. Association for Computational Linguistics.

Jun Yan, Nasser Zalmout, Yan Liang, Christan Grant, Xiang Ren, and Xin Luna Dong. 2021. AdaTag: Multi-attribute value extraction from product profiles with adaptive decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4694–4705, Online. Association for Computational Linguistics.

Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. OpenTag: Open attribute value extraction from product profiles. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1049–1058, London, United Kingdom. ACM.

Tiangang Zhu, Yue Wang, Haoran Li, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2020. Multimodal joint attribute prediction and value extraction for E-commerce product. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2129–2139, Online. Association for Computational Linguistics.

A Appendix

A.1 How to Convert Tuples to Labeled Data

Let's say, we have a tuple of ⟨product title, attribute, value⟩ = ⟨*golf clubs putter pu neutral golf grip, material, pu*⟩, and try to obtain beginning and ending positions of the value in the title. First, we tokenize both title and value using BertTokenizer, and then find a partial token sequence of the title that exactly matches with the token sequence of the value. By performing the match over the tokenization results, we can avoid matching a part of tokens in the title to the value. In case of this example, we can prevent the value *pu* from matching to the first two characters of *putter*. As a result, the value *pu* matches to the token *pu* in the title, and we properly obtain the beginning and ending positions of *pu* in the title.

A.2 Implementation Details

We implemented all the models used in our experiments using PyTorch (Paszke et al., 2019) (ver. 1.7.1),[7] and used "bert-base-uncased" in Transformers (Wolf et al., 2020)[8] as the pre-trained BERT (BERT$_{\text{BASE}}$). The dimension of the hidden states (D) is 768, and the maximum token length of the product title is 64. We set the maximum token length of the query to 32 for all models with the exception of models with the query expansion. To make as many attribute values as possible, we set 192 to the maximum token length of the query for the models using the query expansion, and truncate the concatenated string if the length exceeds 192. We set a rate of dropout over values to 0.2. The total number of parameters in BERT-QA with our query expansion is 109M. We train the models five times with varying random seeds, and average the results.

Regarding to AVEQA, the loss of the distilled masked language model got NaN if we followed the algorithm in the paper. We instead used BERTMLMHead class implemented in Transformers.[8]

We use Adam (Kingma and Ba, 2015) with a learning rate of 10^{-5} as the optimizer. We trained the models up to 20 epochs with a batch size of 32 and chose the models that perform the best micro F_1 on the development set for the test set evaluation.

A.3 Training Time

We used an NVIDIA Quadro M6000 GPU on a server with an Intel® Xeon® E5-2643 v4 3.40GHz CPU with 512GB main memory for training. It took around two hours per epoch for training BERT-QA with our query expansion, while it took around 25 minutes per epoch for training the BERT-QA.

A.4 Preliminary experiments using external resource to obtain the value knowledge

As we have discussed in § 3.1, we can utilize external resource other than the training data of the model to perform the query expansion. We here evaluate the BERT-QA models that have been already trained with our query expansion, using the development data as external (additional) resource to obtain the value knowledge in testing. If new values are retrieved from the development data, the models will build longer queries for attributes. We

[7] https://github.com/pytorch/pytorch/

[8] https://huggingface.co/models

Models		Macro			Micro	
	P (%)	R (%)	F_1	P (%)	R (%)	F_1
			Seen attributes (seen values)			
BERT-QA (Wang et al., 2020)	73.10 (±3.99)	66.86 (±2.97)	69.83 (±3.42)	95.50 (±0.13)	92.14 (±0.21)	93.79 (±0.10)
w/ values in the training data						
BERT-QA +vals	**87.66** (±1.21)	**82.60** (±1.13)	**85.06** (±1.15)	95.76 (±0.21)	92.54 (±0.12)	94.13 (±0.15)
BERT-QA +vals +drop	87.28 (±0.65)	81.83 (±0.90)	84.47 (±0.70)	**95.84** (±0.19)	92.81 (±0.36)	94.30 (±0.17)
BERT-QA +vals +mixing	86.98 (±0.91)	81.36 (±1.13)	84.07 (±0.88)	**95.84** (±0.16)	92.80 (±0.25)	94.29 (±0.10)
BERT-QA +vals +drop +mixing	86.43 (±0.98)	81.48 (±0.70)	83.88 (±0.78)	95.79 (±0.20)	**93.12** (±0.11)	**94.44** (±0.15)
w/ values in the training and development data						
BERT-QA +vals	86.71 (±1.14)	81.50 (±0.83)	84.02 (±0.96)	95.44 (±0.23)	92.00 (±0.13)	93.69 (±0.18)
BERT-QA +vals +drop	85.29 (±1.04)	80.29 (±0.94)	82.71 (±0.93)	95.44 (±0.20)	92.44 (±0.31)	93.92 (±0.12)
BERT-QA +vals +mixing	85.89 (±1.50)	80.51 (±2.26)	83.11 (±1.85)	95.77 (±0.16)	92.77 (±0.22)	94.25 (±0.08)
BERT-QA +vals +drop +mixing	85.65 (±0.57)	80.72 (±0.69)	83.11 (±0.56)	95.75 (±0.16)	93.06 (±0.12)	94.39 (±0.12)
			Seen attributes (unseen values)			
BERT-QA (Wang et al., 2020)	**29.72** (±2.20)	**24.72** (±1.58)	**26.99** (±1.83)	34.44 (±3.47)	**21.28** (±1.80)	**26.28** (±2.26)
w/ values in the training data						
BERT-QA +vals	16.89 (±1.49)	12.94 (±1.46)	14.65 (±1.48)	31.56 (±2.40)	12.42 (±1.15)	17.83 (±1.55)
BERT-QA +vals +drop	22.32 (±1.48)	18.77 (±1.06)	20.39 (±1.24)	37.06 (±1.09)	16.99 (±0.70)	23.30 (±0.81)
BERT-QA +vals +mixing	24.10 (±1.45)	18.98 (±0.90)	21.23 (±1.09)	35.07 (±1.51)	16.77 (±1.02)	22.68 (±1.22)
BERT-QA +vals +drop +mixing	27.19 (±1.20)	22.31 (±1.00)	24.51 (±1.06)	36.60 (±0.50)	18.33 (±0.73)	24.42 (±0.64)
w/ values in the training and development data						
BERT-QA +vals	24.03 (±1.81)	17.94 (±1.56)	20.54 (±1.68)	36.54 (±2.25)	15.71 (±1.31)	21.97 (±1.67)
BERT-QA +vals +drop	27.27 (±1.09)	22.30 (±1.24)	24.53 (±1.17)	**39.49** (±2.31)	19.16 (±0.59)	25.79 (±0.91)
BERT-QA +vals +mixing	27.52 (±1.02)	21.56 (±1.02)	24.17 (±0.97)	37.35 (±1.02)	18.77 (±0.97)	24.98 (±1.02)
BERT-QA +vals +drop +mixing	28.57 (±1.11)	23.44 (±1.13)	25.75 (±1.12)	37.64 (±0.96)	19.67 (±0.60)	25.83 (±0.67)
			Unseen attributes			
BERT-QA (Wang et al., 2020)	42.22 (±6.67)	42.22 (±6.67)	42.22 (±6.67)	59.23 (±8.78)	45.26 (±6.32)	51.22 (±7.14)
w/ values in the training data						
BERT-QA +vals	15.56 (±2.22)	15.56 (±2.22)	15.56 (±2.22)	64.00 (±9.70)	14.74 (±2.11)	23.91 (±3.30)
BERT-QA +vals +drop	19.44 (±2.48)	18.33 (±1.36)	18.85 (±1.85)	56.67 (±9.33)	18.95 (±2.58)	28.37 (±3.98)
BERT-QA +vals +mixing	42.22 (±4.44)	42.22 (±4.44)	42.22 (±4.44)	61.69 (±3.15)	45.26 (±4.21)	52.03 (±2.92)
BERT-QA +vals +drop +mixing	42.22 (±2.72)	42.22 (±2.72)	42.22 (±2.72)	54.42 (±2.48)	45.26 (±2.58)	49.41 (±2.51)
w/ values in the training and development data						
BERT-QA +vals	37.78 (±2.22)	37.78 (±2.22)	37.78 (±2.22)	69.33 (±1.33)	35.79 (±2.11)	47.19 (±2.17)
BERT-QA +vals +drop	43.33 (±2.22)	43.33 (±2.22)	43.33 (±2.22)	72.18 (±1.09)	41.05 (±2.11)	52.32 (±2.02)
BERT-QA +vals +mixing	50.00 (±3.51)	50.00 (±3.51)	50.00 (±3.51)	**74.67** (±2.88)	52.63 (±3.33)	61.69 (±2.86)
BERT-QA +vals +drop +mixing	**52.22** (±2.72)	**52.22** (±2.72)	**52.22** (±2.72)	73.38 (±4.20)	**54.74** (±2.58)	**62.66** (±2.80)

Table 6: Performance on seen and unseen attributes in Table 4 whose new values are retrieved from the development data and are used for the query expansion; reported numbers are mean (std. dev.) of five trials.

here evaluate such attributes with longer queries among the seen and unseen attributes in Table 4.

Table 6 shows the performance of the BERT-QA models with our query expansion on 288 seen values for 107 seen attributes, 339 unseen values for 131 seen attributes, and 19 values for 18 unseen attributes, for which new values are retrieved from the development data. We can observe that the new values retrieved from the development data boosted the performance of the BERT-QA models with our query expansion on the unseen values for the seen attributes and the unseen attributes, whereas they did not increase the performance on the seen values for the seen attributes. In the future, we will explore a better way to leverage the value knowledge in the external resources other than the training data of the QA-based models.

Event-Event Relation Extraction using Probabilistic Box Embedding

EunJeong Hwang, Jay-Yoon Lee, Tianyi Yang, Dhruvesh Patel, Dongxu Zhang & Andrew McCallum
College of Information and Computer Science, University of Massachusetts Amherst
{ehwang,jaylee,tianyiyang,dhruveshpate,
dongxuzhang,mccallum}@cs.umass.edu

Abstract

To understand a story with multiple events, it is important to capture the proper relations across these events. However, existing event relation extraction (ERE) frameworks regard it as a multi-class classification task, and do not guarantee any coherence between different relation types. For instance, if a phone line *died* after *storm*, then it is evident that the *storm* happened before the *died*. Current frameworks of event relation extraction do not guarantee this anti-symmetry and thus enforce it via a constraint loss function (Wang et al., 2020). In this work, we propose to modify the underlying ERE model to guarantee coherence by representing each event as a box representation (BERE) without applying explicit constraints. Our experiments show that BERE has stronger conjunctive constraint satisfaction while performing on par or better in terms of F_1 compared to previous models with constraint injection.[1]

1 Introduction

A piece of text can contain several events. In order to truly understand this text, it is vital to understand the subevent and temporal relationships between these events.(Mani et al., 2006a; Chambers and Jurafsky, 2008; Yang and Mitchell, 2016; Araki et al., 2014). Both temporal as well as subevent relationships between events satisfy transitivity constraints. For instance, in the paragraph, "There was a *storm* in Atlanta in the night. All the phone lines were *dead* the next morning. I was not able to *call* for help.", the event marked by *dead* occurs after *storm* and the event *call* occurs after *dead*. Hence, by transitivity, a sensible model should predict that *storm* occurs before *call*. In general, predicting the relationships between different events in the same document, such that these predictions are coherent, is a challenging task (Xiang and Wang, 2019).

While previous works utilizing neural methods provide competitive performances, these works employ multi-class classification per event-pair independently and are not capable of preserving logical constraints among relations, such as asymmetry and transitivity, during training time (Ning et al., 2019; Han et al., 2019a). To address this problem Wang et al. (2020) introduced a constrained learning framework, wherein they enforce logical coherence amongst the predicted event types through extra loss terms. However, since the coherence is enforced in a soft manner using extra loss terms, there is still room for incoherent predictions. In this work, we show that it is possible to induce coherence in a much stronger manner by representing each event using a box (Dasgupta et al., 2020).

We propose a Box Event Relation Extraction (BERE) model that represents each event as a probabilistic box. Box embeddings (Vilnis et al., 2018) were first introduced to embed nodes of hierarchical graphs into Euclidean space using hyperrectangles, which were later extended to jointly embed multi-relational graphs and perform logical queries (Patel et al., 2020; Abboud et al., 2020). In this paper, we represent an event complex using boxes–one box for each event. Such a model enforces logical constraints by design (see Section 3.2). Consider the example in Figure 1. Event *dead* (e_2) follows event *storm* (e_1), indicating e_2 is child of e_1. Boxes can represent these two events as separate representations and by making e_1 to contain the box e_2, which not only preserve their semantics, but also can infer its antisymmetric relation that event e_1 is a parent of event e_2. However, the previous models based on pairwise-event vector representations have no real relation between representations (e_1, e_2) and (e_2, e_1) that can guarantee the logical coherence.

Experimental results over three datasets, HiEve, MATRES, and Event StoryLine (ESL), show that our method improves the baseline (Wang et al.,

[1] The code is available at https://github.com/iesl/CE2ERE

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 235 - 244
May 22-27, 2022 ©2022 Association for Computational Linguistics

2020) by 6.8 and 4.2 F_1 points on single task and by 0.95 and 3.29 F_1 points on joint task over symmetrical dataset. Furthermore, our BERE model decreases conjunctive constraint violation rate by 85∼88% on a single-task models compared to plain vector model, and by 38% on joint-task model compared to constraint-injected vector model. We show that handling antisymmetric constraints, that exist among different relations, can satisfy the interwined conjunctive constraints and encourage the model towards a coherent output across temporal and subevent tasks.

2 Background

Task description Given a document consisting of multiple events $e_1, e_2, \ldots, e_n$, we wish to predict the relationship between each event pair (e_i, e_j). We denote by $r(e_i, e_j)$ the relation between event pair $(e_i,\ e_j)$. Its values are defined in the label space {PARENT-CHILD, CHILD-PARENT, COREF, NOREL} for subevent relationship (HiEve) and {BEFORE, AFTER, EQUAL, VAGUE} for temporal relationship (MATRES).[2] Both subevent and temporal relationships have four similar-category relationship labels where the first two labels, (PARENT-CHILD,CHILD-PARENT) and (BEFORE, AFTER) hold reciprocal relationship, the third label (COREF and EQUAL) occurs when it is hard to tell which of the first two labels that event pair should be classified to. Lastly, the last label (NOREL and VAGUE) represents a case when an event pair is not related at all.

Box embeddings A box $b = \prod_{i=1}^{d} [b_{m,i}, b_{M,i}]$ such that $b \subseteq R^d$ is characterized by its min and max endpoints $b_m, b_M \in \mathbb{R}^d$, with $b_{m,i} < b_{M,i}\ \forall i$. In the probabilistic gumbel box, these min and max points are taken to be independent gumbel-max and gumbel-min random variables, respectively. As shown in Dasgupta et al. (2020), if b and c are two such gumbel boxes then their volume and intersection is given as:

$$\text{Vol}(b) = \prod_{i=1}^{d} \log\left(1 + \exp\left(\frac{b_{M,i} - b_{m,i}}{\beta} - 2\gamma\right)\right)$$

$$b \cap c = \prod_{i=1}^{d} \left[l(b_{m,i}, c_{m,i}; \beta), l(b_{M,i}, c_{M,i}; -\beta)\right],$$

where $l(x, y; \beta) = \beta \log(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}})$, β is the temperature, which is a hyperparameter, and γ is the Euler-Mascheroni constant.[3]

Logical constraints We define symmetry and conjunction constraints of relations. Symmetry constraints indicate the event pair with flipping orders will have the reversed relation. For example, if $r(e_i, e_j)$ = PARENT-CHILD (BEFORE), then $\tilde{r}(e_j, e_i)$ = CHILD-PARENT (AFTER). Given any two events, e_i and e_j, the symmetry consistency is defined as follows:

$$\bigwedge_{e_i, e_j \in \mathcal{E}, r \in \mathcal{R}_{\mathcal{S}}} r(e_i, e_j) \leftrightarrow \tilde{r}(e_j, e_i) \qquad (1)$$

where r is the relation between events, the $\mathcal{E}$ is the set of all possible events and the $\mathcal{R}_{\mathcal{S}}$ is the set of relations, in which symmetry constraints hold.

Conjunctive constraints refer to the constraints that exist in the relations among any event triplet. The conjunctive constraint rules indicate that given any three event pairs, (e_i, e_j), (e_j, e_k), and (e_i, e_k), then the relation of (e_i, e_k) has to fall into the conjunction set specified based on (e_i, e_j) and (e_j, e_k) pairs (see Appendix Table 6). The conjunctive consistency can be defined as:

$$\bigwedge_{\substack{e_i, e_j, e_k \in \mathcal{E} \\ r_1, r_2 \in \mathcal{R}, r_3 \in \mathcal{D}(r_1, r_2)}} r_1(e_i, e_j) \wedge r_2(e_j, e_k) \rightarrow r_3(e_i, e_k)$$

$$\bigwedge_{\substack{e_i, e_j, e_k \in \mathcal{E} \\ r_1, r_2 \in \mathcal{R}, r_3' \notin \mathcal{D}(r_1, r_2)}} r_1(e_i, e_j) \wedge r_2(e_j, e_k) \rightarrow \neg r_3'(e_i, e_k)$$

where the $\mathcal{E}$ is the set of all possible events, r_1 and r_2 are any possible relations exist in the set of all relations $\mathcal{R}$, r_3 is the relation, which is specified by r_1 and r_2 based on conjunctive induction table, and $\mathcal{D}$ is the set of all possible relations, in which r_1 and r_2 have no conflicts in between. The full explanation on symmetry and conjunction consistency can be found in Wang et al. (2020).

3 BERE model

In this section, we present the proposed box model BERE for event-event relation extraction. As depicted in Figure 1, the proposed model encodes each event e_i as a box b_i in $\mathbb{R}^d$ based on e_i's contextualized vector representation h_i. As described in §3.1, the relation between (e_i, e_j) is then predicted using conditional probability scores $P(b_i|b_j) = \text{Vol}(b_i \cap b_j)/\text{Vol}(b_j)$, $P(b_j|b_i) = \text{Vol}(b_i \cap b_j)/\text{Vol}(b_i)$ defined on box space. Lastly, §3.2 describes loss function used to learn the parameters of the model.

[2] See Experimental Setup 4.1 for the detailed information of HiEve and Matres.

[3] https://en.wikipedia.org/wiki/Euler%27s_constant

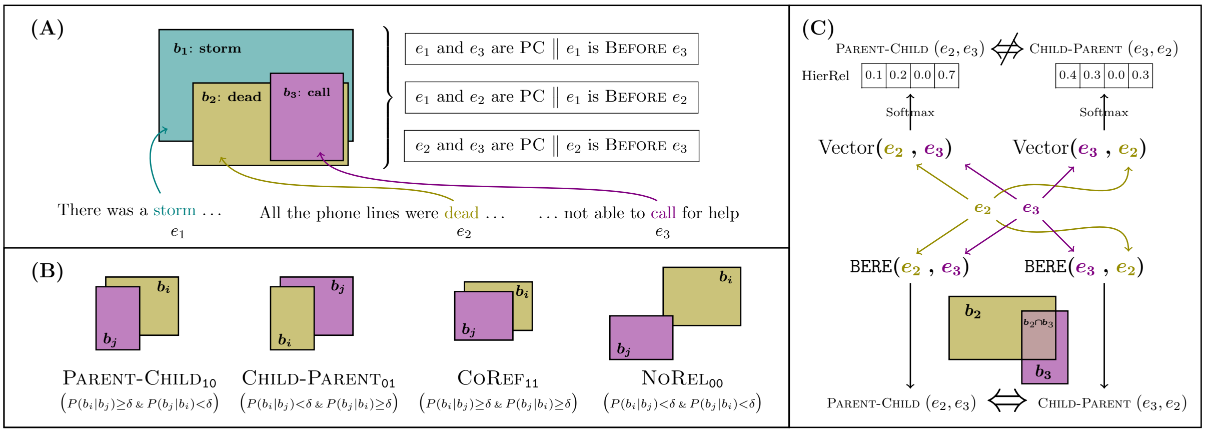

(A)
b1: storm
b2: dead
b3: call
e1 and e3 are PC ‖ e1 is BEFORE e3
e1 and e2 are PC ‖ e1 is BEFORE e2
e2 and e3 are PC ‖ e2 is BEFORE e3
There was a storm . . .
e1
All the phone lines were dead . . .
e2
. . . not able to call for help
e3
(B)
bi
bj
PARENT-CHILD10
(P(bi|bj)≥δ & P(bj|bi)<δ)
CHILD-PARENT01
(P(bi|bj)<δ & P(bj|bi)≥δ)
COREF11
(P(bi|bj)≥δ & P(bj|bi)≥δ)
NOREL00
(P(bi|bj)<δ & P(bj|bi)<δ)
(C)
PARENT-CHILD (e2, e3) ⇎ CHILD-PARENT (e3, e2)
HierRel
0.1 0.2 0.0 0.7
0.4 0.3 0.0 0.3
Softmax
Vector(e2 , e3)
Vector(e3 , e2)
e2
e3
BERE(e2 , e3)
BERE(e3 , e2)
b2
b2∩b3
b3
PARENT-CHILD (e2, e3) ⇔ CHILD-PARENT (e3, e2)

Figure 1: (A) BOX model architecture. (B) Mapping from box positions to event relations with classification rule below. (C) An example shows the fundamental difference between VECTOR and BOX model: BOX model will map events into consistent box representations regardless of the order; VECTOR model treats both cases separately and may not persist logical consistency.

3.1 Inference rule on conditional probability

Notice that given two boxes b_i and b_j, a higher value of $P(b_i|b_j)$ (resp. $P(b_j|b_i)$) implies that box b_j is contained in b_i (resp. b_i contained in b_j). Moreover, other than complete containment in either direction, there are other two prominent configurations possible, i.e. one where b_i, b_j overlap but none contains the other, and the one where b_i, b_j do not overlap. It is possible to capture all four configurations by comparing the values of $P(b_i|b_j)$ and $P(b_j|b_i)$ with a threshold δ. Figure 1(B) states our classification rule formulated based on this observation. With this formulation we have the desired symmetry constraint, i.e., $r(e_i, e_j) = \text{PARENT-CHILD} \iff r(e_j, e_i) = \text{CHILD-PARENT}$, satisfied by design.

3.2 Loss functions for training

BCE loss As we require two dimensions of scalar $P(b_i|b_j)$ and $P(b_j|b_i)$ to classify $r(e_i, e_j)$, and for ease of notation, we define our label space with 2-dimensional binary variable $y^{(i,j)}$ as shown in Figure1(b). Where $y_0^{(i,j)} = I(P(b_i|b_j) \geq \delta)$ and $y_1^{(i,j)} = I(P(b_j|b_i) \geq \delta)$ where $I(\cdot)$ stands for indicator function. Now given batch B, BCE loss ($\mathbf{L}_1$) is defined as:

$$- \sum_{(i,j)\in B} y_0^{(i,j)} \ln P(b_i|b_j) + (1 - y_0^{(i,j)}) \ln (1 - P(b_i|b_j))$$
$$+ y_1^{(i,j)} \ln P(b_j|b_i) + (1 - y_1^{(i,j)}) \ln (1 - P(b_j|b_i)).$$

Pairwise loss Motivated from previous papers using pairwise features to characterize relations, we also incorporate a pairwise box into our learning objective, and only in learning time, to encourage relevant boxes to be concentrated together. For the event-pair representation, two contextualized event embeddings (h_i, h_j) are combined as $[h_i, h_j, h_i \odot h_j]$ where $\odot$ represents element-wise multiplication. Then, a multi-layer perceptron (MLP) is used to transform pairwise vectors to box representations b_{ij}. The pairwise features we use here are similar to (Zhou et al., 2020) except that we do not use subtraction in order to preserve symmetry between pairwise features of (e_i, e_j) and (e_j, e_i), i.e. $b_{ij} = b_{ji}$. For two related events, we enforce the intersection of corresponding boxes $b_i \cap b_j$ to be inside the pairwise box. For irrelevant event pairs such as having NOREL or VAGUE, their intersection and pairwise boxes are forced to be disjoint. The pairwise loss $\mathbf{L}_2$ is defined as:

$$- \sum_{i,j\in R^+} \log P(b_i \cap b_j | b_{ij}) - \sum_{i,j\in R^-} \log \Big(1 - P(b_i \cap b_j | b_{ij})\Big)$$

where R^- is a set of irrelevant relations, such as NOREL and VAGUE, and R^+ stands for complement set of R^-, i.e. all the set of relations that indicates two events have some relation.

In the remainder of the paper, `BERE` refers to a model trained with loss $\mathbf{L}_1$ and `BERE-p` refers to a model trained with two losses $\mathbf{L}_1, \mathbf{L}_2$ combined.

4 Experiments

In this section, we describe datasets, baseline methods, and evaluation metrics. Lastly, we provide experimental results and a detailed analysis of logical consistency.

4.1 Experimental Setup

Datasets Experiments are conducted over three asymmetrical event relation extraction corpus,

Table 1: An overview of dataset statistics.

	HiEve	MATRES	ESL
# of Documents			
Train	80	183	155
Dev	-	72	51
Test	20	20	52
# of Pairs			
Train	35001	6332	2238
Test	7093	827	619
# of Pairs for Symmetrical Dataset			
Test	8693	1493	1222

Table 2: Mapped relation labels from ESL to HiEve

Original labels in ESL	Mapped Labels
RISING_ACTION	Parent-Child
CONTAINS	
BEFORE	
PRECONDITION	
ENDED_ON	
FALLING_ACTION	Child-Parent
AFTER	
BEGUN_ON	
CAUSE	
OVERLAP	NoRel

HiEve (Glavaš and Šnajder, 2014), MATRES (Ning et al., 2018), and Event StoryLine (ESL) (Caselli and Vossen, 2017). Table 1 shows a brief summary of dataset statistics. HiEve consists of 100 articles and the narratives in news stories are represented as event hierarchies. The annotations include subevent and coreference relations. MATRES is a four-class temporal relation dataset, which contains 275 news articles drawn from a number of different sources. Event StoryLine (ESL) corpus is a dataset that contains 258 news documents and includes event temporal and subevent relations. The ESL dataset is defined differently compared to HiEve and MATRES, so we mapped the ESL labels into the labels in HiEve similar to (Wang et al., 2020) as shown in Table 2.

For creating symmetrical dataset, we augment Parent-Child and Child-Parent (Before and After) pairs by their reversed relations Child-Parent and Parent-Child (After and Before), respectively.

Baseline We compare our BERE, BERE-p against the state-of-the-art event-event relation extraction model proposed by (Wang et al., 2020). This model utilizes RoBERTa with frozen parameters and further trains BiLSTM to represent text inputs into vector h_i (for e_i) and then further utilizes MLP to represent pairwise representation v_{ij} for (e_i, e_j). Given v_{ij}, vector model (Vector) simply computes softmax over projected logits to produce probability for every possible relations. On top of this, as (Wang et al., 2020) showed that constraint injection improves performance, we also compare with the constraint-injected model (Vector-c).

Table 3: F_1 scores of BERE and BERE-p

Model	F_1 Score	
	HiEve	MATRES
BERE	0.4483	0.7069
BERE-p	0.4771	0.7105

For a fair comparison, we utilize the same RoBERTa + BiLSTM + MLP architecture for projecting event to box representation.

Metrics Following the same evaluation setting in previous works, we report the micro-F_1 score of all pairs, except Vague pairs, on MATRES (Han et al., 2019b; Wang et al., 2020). On HiEve and ESL, the micro-F_1 score of Parent-Child and Child-Parent pairs is reported (Glavaš and Šnajder, 2014; Wang et al., 2020).

4.2 Results and Discussion

Impact of pairwise box, Table 3 We first show the results of the BERE and BERE-p with and without pairwise loss. The model with pairwise loss shows about 2.8 F_1 point improvement on HiEve and 1 F_1 point improvement on MATRES. It indicates that promoting the relevant event pairs to mingle together in the geometrical space is helpful and it is particularly useful when most of the relation extraction model encodes individual sentences independently.

Vector-based vs. Box-based, Table 4 Table 4 shows a comparison of our box approach to the baseline with the ratio of symmetric and conjunctive constraint violations. Our approach clearly outperforms the baseline methods on symmetric evaluation with a gain of 6.79, 4.26, and 9.34 F_1 points on the single task over HiEve, MATRES, and ESL datasets, respectively and with a gain of 0.95 and 3.29 F_1 points on the joint task over HiEve and MATRES. The performance gains from asymmetrical to symmetrical datasets with BERE-p are much larger compared to the increase of Vectors. This demonstrates the BERE-p successfully captures symmetrical relations, while previous vector models do not. In addition, it is noteworthy that our method without constrained learning excels Vector-c, which is trained with constrained learning. This suggests that the inherent ability to

Table 4: F_1 scores with symmetric and conjunctive constraint violation results over original and symmetrical datasets. symm const. and conj const. denote symmetric and conjunctive constraint violations (%), respectively; H, M, and ESL are HiEve, MATRES, Event StoryLine datasets, respectively; single task (top) and joint task (bottom)

Model	F_1 Score						symmetry const.			conjunctive const.		
	Original data			Symmetric evaluation								
	H	M	ESL	H	M	ESL	H	M	ESL	H	M	ESL
`Vector`	0.4437	**0.7274**	0.2660	0.5385	0.7288	0.4444	22.49	35.81	60.9	4.91	2.53	6.1
`BERE-p`	**0.4771**	0.7105	**0.3214**	**0.6064**	**0.7714**	**0.5379**	**0**	**0**	**0**	**0.71**	**0.30**	**0**
	Joint model (H&M)						Joint eval (H&M)			Joint eval (H&M)		
`Vector`	0.4727	**0.7291**	n/a	0.5517	0.7405	n/a	24.08		n/a	6.17		n/a
`Vector-c`	**0.5262**	0.7068		0.6166	0.7106		28.83			2.98		
`BERE-p`	0.5053	0.7125		**0.6261**	**0.7734**		**0**			**1.84**		

Table 5: Symmetric evaluation of `Vector`* (`Vector` trained with reciprocal dataset) and `BERE-p` trained with original dataset on joint task. s-const. and c-const. denote symmetric and conjunctive constraint violations (%), respectively

Model	F_1 Score			
	HiEve	MATRES	s-const.	c-const.
`Vector`*	0.6120	0.7720	12.01	6.70
`BERE-p`	**0.6261**	**0.7734**	**0**	**1.84**

model symmetrical relations helps satisfy the intertwined conjunctive constraints, thus producing more coherent results from a model. See Appendix E for constraint violation statistics for asymmetric dataset.

Constraint Violation Analysis, Table 8 (Appendix) We analyze constraint violations for each label from both HiEve and MATRES. For label pairs from the same dataset, our approach excels in almost every cases. For label pairs across datasets, our approach also shows fewer or similar levels of violation. This further indicates, without explicitly injecting constraints into objectives, our model can persist logical consistency among different relations.

5 Ablation Study

We conduct additional experiments to see whether `Vector` trained with the augmented symmetrical dataset will affect the conclusion of `BERE-p`. The results in Table 5 reconfirm the `BERE-p`'s superior ability in handling constraints with better performance, while `Vector` requires significantly longer training time due to the extended training dataset with worse performance. We also note that training `Vector` with the augmented symmetrical dataset does not help with conjunctive constraint violations (6.17 → 6.70), although it reduces symmetrical constraint violations (24.08 → 12.01).

6 Conclusion

We propose a novel event relation extraction method that utilizes box representation. The proposed method projects each event to a box representation which can model asymmetric relationships between entities. Utilizing this box representation, we design our relation extraction model to handle antisymmetry between events of (e_i, e_j) and (e_j, e_i) which previous vector models were not capable of. Through experiments on three datasets, we show that the proposed method not only free of antisymmetric constraint violations but also have drastically lower conjunctive constraint violations while maintaining similar or better performance in F_1. Our model shows that box representation can provide coherent classification across multiple event relations and opens up future research for box representations in event-to-event relation classification.

7 Acknowledgement

This work is based upon work supported in part by the Center for Data Science and the Center for Intelligent Information Retrieval, and in part by the National Science Foundation under Grant Nos. 1763618 and 1514053, and in part by the Chan Zuckerberg Initiative under the project "Scientific Knowledge Base Construction". Some of the work reported here was performed using high performance computing equipment obtained under a grant from the Collaborative R&D Fund managed by the Massachusetts Technology Collaborative.

References

Ralph Abboud, İsmail İlkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. Boxe: A box embedding model for knowledge base completion. In *Proceedings of the Thirty-Fourth Annual Conference on Advances in Neural Information Processing Systems (NeurIPS)*.

Jun Araki, Zhengzhong Liu, Eduard Hovy, and Teruko Mitamura. 2014. Detecting subevent structure for event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

Tommaso Caselli and Piek Vossen. 2017. The event StoryLine corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop*, pages 77–86, Vancouver, Canada. Association for Computational Linguistics.

Nathanael Chambers and Daniel Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 698–706, Honolulu, Hawaii. Association for Computational Linguistics.

Tejas Chheda, Purujit Goyal, Trang Tran, Dhruvesh Patel, Michael Boratko, Shib Sankar Dasgupta, and Andrew McCallum. 2021. Box embeddings: An open-source library for representation learning using geometric structures. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 203–211, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shib Sankar Dasgupta, Michael Boratko, Dongxu Zhang, Luke Vilnis, Xiang Lorraine Li, and Andrew McCallum. 2020. Improving local identifiability in probabilistic box embeddings. In *NeurIPS*.

Dmitriy Dligach, Timothy Miller, Chen Lin, Steven Bethard, and Guergana Savova. 2017. Neural temporal relation extraction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 746–751, Valencia, Spain. Association for Computational Linguistics.

Goran Glavaš and Jan Šnajder. 2014. Constructing coherent event hierarchies from news stories. In *Proceedings of TextGraphs-9: the workshop on Graph-based Methods for Natural Language Processing*, pages 34–38, Doha, Qatar. Association for Computational Linguistics.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019a. Joint event and temporal relation extraction with shared representations and structured prediction. In *2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019b. Joint event and temporal relation extraction with shared representations and structured prediction. *CoRR*, abs/1909.05360.

Inderjeet Mani, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. 2006a. Machine learning of temporal relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 753–760, Sydney, Australia. Association for Computational Linguistics.

Inderjeet Mani, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. 2006b. Machine learning of temporal relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, page 753–760, USA. Association for Computational Linguistics.

Qiang Ning, Zhili Feng, and Dan Roth. 2017. A structured learning approach to temporal relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1027–1037, Copenhagen, Denmark. Association for Computational Linguistics.

Qiang Ning, Sanjay Subramanian, and Dan Roth. 2019. An improved neural baseline for temporal relation extraction. In *EMNLP*.

Qiang Ning, Hao Wu, and Dan Roth. 2018. A multi-axis annotation scheme for event temporal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1318–1328, Melbourne, Australia. Association for Computational Linguistics.

Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. 2021. Modeling fine-grained entity types with box embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2051–2064, Online. Association for Computational Linguistics.

Dhruvesh Patel, Pavitra Dangati, Jay-Yoon Lee, Michael Boratko, and Andrew McCallum. 2022. Modeling label space interactions in multi-label classification using box embeddings. In *International Conference on Learning Representations*.

Dhruvesh Patel, Shib Sankar Dasgupta, Michael Boratko, Xiang Li, Luke Vilnis, and Andrew McCallum. 2020. Representing joint hierarchies with box embeddings. In *Automated Knowledge Base Construction*.

Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, page 75–80, USA. Association for Computational Linguistics.

Marc Verhagen and James Pustejovsky. 2008. Temporal processing with the tarsqi toolkit. In *22nd International Conference on on Computational Linguistics: Demonstration Papers*, COLING '08, page 189–192, USA. Association for Computational Linguistics.

Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. 2018. Probabilistic embedding of knowledge graphs with box lattice measures. In *ACL*. Association for Computational Linguistics.

Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. 2020. Joint constrained learning for event-event relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 696–706. Association for Computational Linguistics.

Wei Xiang and Bang Wang. 2019. A survey of event extraction from text. *IEEE Access*, 7:173111–173137.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 289–299, San Diego, California. Association for Computational Linguistics.

Guangyu Zhou, Muhao Chen, Chelsea J T Ju, Zheng Wang, Jyun-Yu Jiang, and Wei Wang. 2020. Mutation effect estimation on protein–protein interactions using deep contextualized representation learning. *NAR Genomics and Bioinformatics*, 2(2). Lqaa015.

A Hyperparameters

We utilize 768 dimensional pretrained RoBERTa model to compute word embeddings for events. models are trained for 100 epochs with AMSGrad optimizer and the learning rate is set to be 0.001. On HiEve and ESL, we sample NOREL in trainset using downsample ratio, which is fixed to 0.015, and the downsample ratio for valid and test sets is fixed to 0.4. This is to encourage the models to learn and evaluate all types of relations that exist in the datasets when NOREL overwhelmingly represents the dataset. We use three weights, λ_1, λ_2, and λ_3, to balance our three learning objectives L_1, L_2, and L_3 (see Section 3.2 and Appendix B), in which the weights are selected between 0.1 and 1. A threshold δ for HiEve is selected between -0.4 and -0.3 and a threshold for MATRES is chosen between -0.7 and -0.6. We use wandb (Biewald, 2020) tool for efficient hyperparameter tuning.

B Conjunctive Consistency Loss

With consistency requirements on conjunctive relations over temporal and subevent datasets (as shown in Table 6), we incorporate the loss function introduced by (Wang et al., 2020) into our box model to handle conjunctive constraints. Three events are grouped into three pairs, $(e1, e2), (e2, e3)$ and $(e1, e3)$, and the relation score for each class is calculated based on conditional probabilities and its binary logits. With the relation labels defined for each class (see Section 3.2), the relation score, $r(e_1, e_2)$, is calculated as:

$$r_i = y_0^{(i,j)} \log P(b_i|b_j) + y_1^{(i,j)} \log P(b_j|b_i) \quad (2)$$

where $y_0^{(i,j)} = I(P(b_i|b_j) \geq \delta)$ and $y_1^{(i,j)} = I(P(b_j|b_i) \geq \delta)$ and $y_0^{(i,j)}$ and $y_1^{(i,j)}$ are the first and second binary logits in relation label, respectively. Using this relation score, we now define the loss function for modeling conjunction constraints:

$$L_3 = \sum |L_{t1}| + \sum |L_{t2}|, \quad (3)$$

where the two transitivity losses are defined as

$$L_{t1} = \log r_{(e1,e2)} + \log r_{(e2,e3)} + \log r_{(e1,e3)}$$
$$L_{t2} = \log r_{(e1,e2)} + \log r_{(e2,e3)} + \log(1 - r_{(e1,e3)})$$

Table 7 presents the results of BERE-p combined with the above learning objective, denoted as BERE-c. Compared to the results from BERE-p, BERE-c shows a significantly smaller ratio of constraint violations than BERE-p, while sacrificing F_1 by ∼2 point from the performance with BERE-p.

C Vector model architecture

Refer to Figure 2 for architecture of previous vector models.

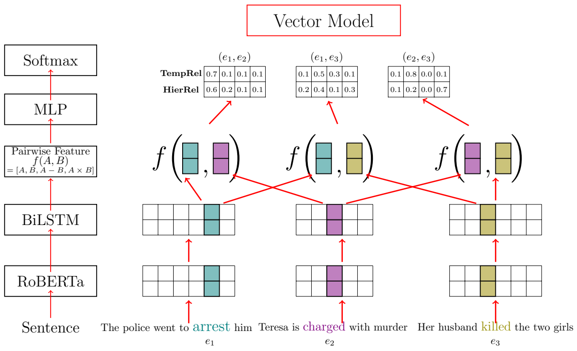


Figure 2: VECTOR model architecture.

D Detailed analysis on conjunctive constraint violation

Constraint Violation Analysis, Table 8 We further break down constraint violations for each label on HiEve and MATRES. The comparison of constraint violations between the vector model with constrained learning (Vector-c) and the box model without constrained learning (BERE-p) is shown in Table 8. "n/a" refers to no predictions and this frequently appears on COREF and EQUAL due to their sparsity in the corpus. Our approach shows a smaller ratio of constraint violations in most of the categories, with only a few exceptions. 2nd and 3rd quadrants (HiEve→MATRES and MATRES→HiEve) stand for cross-category, while 1st and 4th quadrants (HiEve→HiEve and MATRES→MATRES) stand for the same-category. Interestingly, our approach without any injected constraints shows a smaller or similar ratio to Vector-c in the cross-category as well as in the same-category. We calculated $r_c = \frac{\text{total \# of cross-category const-violations}}{\text{total \# of cross-category event triplets}}$ and $r_s = \frac{\text{total \# of same-category const-violations}}{\text{total \# of same-category event triplets}}$, where const-violations refers to constraint violations. r_c for Vector-c is 6.26% and for BERE-p is 4.55% and r_s for Vector-c is 0.05% and for BERE-p is 0.017%. This confirms the

Table 6: The induction table for conjunctive constraints on temporal and subevent relations (Wang et al., 2020). Given three events, $e1$, $e2$, and $e3$, the left-most column is $r_1(e_1, e_2)$ and the top row is $r_2(e_2, e_3)$.

	PC	CP	CR	NR	BF	AF	EQ	VG
PC	PC, -AF	–	PC, -AF	-CP, -CR	BF, -CP, -CR	–	BF, -CP, -CR	–
CP	–	CP, -BF	CP, -BF	-PC, -CR	–	AF, -PC, -CR	AF, -PC, -CR	–
CR	PC, -AF	CP, -BF	CR, EQ	NR	BF, -CP, -CR	AF, -PC, -CR	EQ	VG
NR	-CP, -CR	-PC, -CR	NR	–	–	–	–	–
BF	BF, -CP, -CR	–	BF, -CP, -CR	–	BF, -CP, -CR	–	BF, –CP, –CR	-AF, -EQ
AF	–	AF, -PC, -CR	AF, -PC, -CR	–	–	AF, -PC, -CR	AF, -PC, -CR	-BF, -EQ
EQ	-AF	-BF	EQ	–	BF, -CP, -CR	AF, -PC, -CR	EQ	VG, -CR
VG	–	–	VG, -CR	–	-AF, -EQ	-BF, -EQ	VG	-

Table 7: F_1 scores and the ratio of symmetric and conjunctive constraint violations of box model with constrained learning over `Eval-A` and `Eval-S`; `Eval-A` and `Eval-S` denote asymmetrical and symmetrical evaluation datasets, respectively. const. means constraint violations; results are on joint task.

Model	F_1 Score				symmetry const. (%)		conjunctive const. (%)	
	`Eval-A`		`Eval-S`		`Eval-A`	`Eval-S`	`Eval-A`	`Eval-S`
	HiEve	MATRES	HiEve	MATRES				
`BERE-p`	0.5053	0.7125	0.6261	0.7734	0	0	3.12	1.84
`BERE-c`	0.5083	0.7021	0.6183	0.7562	0	0	0.39	0.19

Table 8: Constraint violation analysis over HiEve and MATRES. See Appendix B for conjunctive consistency requirements; PARENT-CHILD (PC), CHILD-PARENT (CP), COREF (CR), NOREL (NR), BEFORE (BF), AFTER (AF), EQUAL (EQ), VAGUE (VG); "-" means no existing constraint violations; constraint injected vector model (top), box model with using pairwise loss (bottom).

`Vector-c`

	PC	**CP**	**CR**	**NR**	**BF**	**AF**	**EQ**	**VG**
PC	0.05	-	0.13	0.02	0.20	-	0.5	-
CP	-	0.33	0.46	0.01	-	0.25	n/a	-
CR	0.12	0.42	0.68	0.08	0.19	0.43	n/a	0.27
NR	0.01	0.03	0.13	-	-	-	-	-
BF	0.23	-	0.41	-	0.12	-	0.42	0.02
AF	-	0.33	0.30	-	-	0.01	0.13	0.05
EQ	0.00	0.50	n/a	-	0.25	0.00	n/a	0.50
VG	-	-	0.34	-	0.03	0.02	n/a	-

`BERE-p`

	PC	**CP**	**CR**	**NR**	**BF**	**AF**	**EQ**	**VG**
PC	0.13	-	n/a	0.00	0.16	-	0.30	-
CP	-	0.23	n/a	0	-	0.28	0.34	-
CR	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
NR	0.00	0.00	n/a	-	-	-	-	-
BF	0.24	-	n/a	-	0.08	-	0.32	0.00
AF	-	0.17	n/a	-	-	0.05	0.12	0.00
EQ	0.23	0.29	n/a	-	0.15	0.18	n/a	0.00
VG	-	-	n/a	-	0.00	0.00	0.13	-

effectiveness of having boxes in handling logical consistency among different relations.

E Symmetric and conjunctive constraint violations over origianl data

Table 9 shows the F_1 and symmetry and conjunctive constraint violation results over original dataset. The results of symmetry and conjunctive constraint violations confirm our expectation and exhibit a similar observation from Table 4.

F Related Work

F.1 Event-Event Relation Extraction

This task has been traditionally modeled as a pairwise classification task with hand-engineered features and early attempts applied conventional machine learning methods, such as logistic regressions and SVM (Mani et al., 2006b; Verhagen et al., 2007; Verhagen and Pustejovsky, 2008). Later works utilized a structured learning (Ning et al., 2017) and neural methods to characterize relations. The neural methods have been shown effective and ensure logical consistency on relations through inference step (Dligach et al., 2017; Ning et al., 2018, 2019; Han et al., 2019a). More recent works proposed a constrained learning framework, which facilitates constraints during training time (Han et al., 2019b; Wang et al., 2020). Motivated by these works, we propose a box model to automatically handle inherent constraints without heavily relying on constrained learning across two different tasks.

F.2 Box Embeddings

Box embeddings (Vilnis et al., 2018) were introduced as a shallow model to embed nodes of hierarchical graphs into euclidean space using hyperrectangles, which were later extended to jointly embed multi-relational graphs and perform logical queries (Patel et al., 2020; Abboud et al., 2020). Recent works have successfully used box representations in conjunction with neural networks to represent input text for tasks like entity typing (Onoe et al., 2021), multi-label classification (Patel et al., 2022), natural language entailment (Chheda et al., 2021), etc. In all these works, the input is rep-

Table 9: F_1 scores with symmetric and conjunctive constraint violation results over original datasets. symm const. and conj const. denote symmetric and conjunctive constraint violations, respectively; H, M, and ESL are HiEve, MATRES, Event StoryLine datasets, respectively; single task(top) and joint task(bottom)

Model	F1 Score			symmetry const. (%)			conjunctive const.(%)		
	Original data								
	H	M	ESL	H	M	ESL	H	M	ESL
Vector	0.4437	**0.7274**	0.2660	22.73	38.63	56.7	5.66	0.69	9.4
BERE-p	**0.4771**	0.7105	**0.3214**	**0**	**0**	**0**	**0.75**	**0.46**	**0**
Joint				H+M			H+M		
Vector	0.4727	**0.7291**	n/a	23.04		n/a	10.85		n/a
Vector-c	**0.5262**	0.7068		23.83			3.52		
BERE-p	0.5053	0.7125		**0**			**3.12**		

resented using a single box by transforming the output of the neural network into a hyper-rectangle. In this paper, we take this a step forward by representing the input event complex using multiple boxes. Our single box model represents each even in an input paragraph using a box and the pairwise box model adds on top of these, one box each for every pair of events (see section 3.2).

Sample, Translate, Recombine: Leveraging Audio Alignments for Data Augmentation in End-to-end Speech Translation

Tsz Kin Lam[1] and **Shigehiko Schamoni**[2,1] and **Stefan Riezler**[1,2]
[1]Department of Computational Linguistics, Heidelberg University
[2]Interdisciplinary Center for Scientific Computing (IWR), Heidelberg University
{lam,schamoni,riezler}@cl.uni-heidelberg.de

Abstract

End-to-end speech translation relies on data that pair source-language speech inputs with corresponding translations into a target language. Such data are notoriously scarce, making synthetic data augmentation by back-translation or knowledge distillation a necessary ingredient of end-to-end training. In this paper, we present a novel approach to data augmentation that leverages audio alignments, linguistic properties, and translation. First, we augment a transcription by *sampling* from a suffix memory that stores text and audio data. Second, we *translate* the augmented transcript. Finally, we *recombine* concatenated audio segments and the generated translation. Besides training an MT-system, we only use basic off-the-shelf components without fine-tuning. While having similar resource demands as knowledge distillation, adding our method delivers consistent improvements of up to 0.9 and 1.1 BLEU points on five language pairs on CoVoST 2 and on two language pairs on Europarl-ST, respectively.

1 Introduction

End-to-end automatic speech translation (AST) relies on data that consist only of speech inputs and corresponding translations. Such data are notoriously limited. Data augmentation approaches attempt to compensate the scarcity of such data by generating synthetic data by translating transcripts into foreign languages or by back-translating target-language data via text-to-speech synthesis (TTS) (Pino et al., 2019; Jia et al., 2019), or by performing knowledge distillation using a translation system trained on gold standard transcripts and reference translations (Inaguma et al., 2021). In this paper, we present a simple, resource conserving approach that does not require TTS and yields improvements complementary to knowledge distillation (KD).

For training cascaded systems, monolingual data for automatic speech recognition and textual translation data for machine translation can be used, reducing the problem of scarcity. Cascaded systems, however, suffer from error propagation, which has been addressed by using more complex intermediate representations such as n-best machine translation (MT) outputs or lattices (Bertoldi and Federico, 2005; Beck et al., 2019, *inter alia*) or by modifying training data to incorporate errors from automatic speech recognition (ASR) and MT (Ruiz et al., 2015; Lam et al., 2021b). End-to-end systems are unaffected by this kind of error propagation and are able to surpass cascaded systems if trained on sufficient amounts of data (Sperber and Paulik, 2020).

Our approach transfers an idea on aligned data augmentation that has been presented for ASR (Lam et al., 2021a) to aligned data augmentation in AST. Similar to aligned data augmentation for ASR, we utilize forced alignment information to create unseen training pairs in a structured manner. Our augmentation procedure consists of the following steps: (1) *Sampling* of a replacement suffix of a transcription and its aligned speech representations, guided by linguistic constraints. (2) *Translation* of the transcription containing the new suffix. (3) *Recombination* of audio data containing the new suffix and the generated translation to distill a new training pair. We thus use the acronym STR (Sample, Translate, Recombine) to refer to our method.

In comparison to Pino et al. (2019) and Jia et al. (2019) who use TTS to generate synthetic speech, we create new examples by recombining real human speech. This reduces the problem of overfitting to synthetic data as for example in SkinAugment (McCarthy et al., 2020) where synthetic audio is generated by auto-encoding speaker conversions. The basic idea of our method is comparable to data augmentation techniques for images such as CutMix (Yun et al., 2019) where images are blended together to form new data examples. However, CutMix selects images randomly, while we recombine phrases in a structured manner.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 245 - 254
May 22-27, 2022 ©2022 Association for Computational Linguistics

Our experimental evaluation is conducted for five language pairs on the CoVoST 2 dataset (Wang et al., 2021) and for two language pairs on the Europarl-ST (Iranzo-Sánchez et al., 2020) dataset. We find considerable improvements for all language pairs on all datasets for our approach on top of KD. Our approach can be seen as an enhancement of Inaguma et al. (2021)'s KD approach since it requires roughly the same computational resources and consistently improves their gains.

2 Method

Our method exploits audio-transcription alignment information to generate previously unseen data pairs for end-to-end AST training. By applying a Part-of-Speech (POS) Tagger on a sentence, we identify potential "pivoting tokens" where the token's prefix or suffix, i.e., the preceding or succeeding tokens, can be exchanged between other sentences containing the same token of the same syntactic function. We then sample possible suffixes for that token from a suffix memory containing text and audio suffixes, and concatenate the prefix, verb, and suffix to generate a new transcription. Then, an MT system translates the new transcription, picking up on the idea of knowledge distillation in AST (Inaguma et al., 2021). The MT system is trained or fine-tuned on the transcription-translation pairs. Finally, using the previously sampled audio suffix, we concatenate prefix, verb, and suffix audio together with the MT generated translation to recombine a new audio-translation pair for end-to-end AST training.

Our augmentation method implements linguistic constraints by making use of the transcription's syntactic structure in combination with alignment information. Effectively, we exploit the strict SVO-scheme of English sentences as we select the verb as our pivoting token. Our method is applicable to other languages, however, it will require more effort to identify exchangeable syntactic structures.

Figure 1 illustrates our approach. We start by identifying the pivoting token in a transcription we want to augment, here "playing" in the sentence "two children are *playing* on a statue". Then, we extract the list of possible suffixes following "playing" from the suffix memory and sample a single audio-text suffix, here "volleyball in a park". Together with the original prefix and pivoting token, the textual part of the sampled suffix builds a new augmented transcription. Similarly, together with the audio prefix and token, the audio part of the suffix builds a new augmented audio example. The augmented transcription is then translated by an MT model. The new audio example (i.e., the representation of "two children playing volleyball in a park") and the translation (i.e., the text "Zwei Kinder spielen Volleyball in einem Park") are then recombined to form a new audio-translation pair.

3 Experimental Setting

Data Preprocessing We evaluate our method on two common AST datasets, CoVoST 2 (Wang et al., 2021) and Europarl-ST (Iranzo-Sánchez et al., 2020). Since Europarl-ST is too small for MT training from scratch, we use 1.6M En-De sentence pairs from Wikipedia following Schwenk et al. (2021) and 3.2M En-Fr sentence pairs from the Common Crawl corpus[1] as additional data. More details on the datasets are in Appendix A.1.

For speech data preprocessing, we extract log Mel-filter banks of 80 dimensions computed every 10ms with a 25ms window. We normalize the speech features per channel using mean and variance per instance. For all textual data, punctuation is normalized using SACREMOSES.[2] The transcriptions are lowercased with punctuation removed.

For the speech-to-text tasks on CoVoST 2, we employ character-level models due to the availability of pre-trained high quality ASR models. For the speech-to-text tasks on Europarl-ST, we learn 5,000 subword units for each target language. For the machine translation tasks in knowledge distillation, we learn a joint subword vocabulary on both source and target for each language pair of size 5,000 for CoVoST 2 and size 40,000 for Europarl-ST including the additional training data. Subword unit creation is always conducted with SENTENCEPIECE (Kudo and Richardson, 2018).

The Montreal Forced Aligner (McAuliffe et al., 2017) is applied without any fine-tuning to extract the acoustic alignments. Thus, the obtained alignments can be of low quality and we discard such examples from our augmentation procedure. Please refer to Appendix A.2 for details on our filtering criteria. To extract POS-tags, we use the SPACY[3] toolkit. We select the verb as our pivoting token and generate the suffix memory as follows: for each verb, we generate a list of audio-text suffix

[1] www.statmt.org/wmt13/..., accessed 3/11/2022

[2] github.com/alvations/sacremoses, accessed 3/11/2022

[3] github.com/explosion/spaCy, accessed 3/11/2022

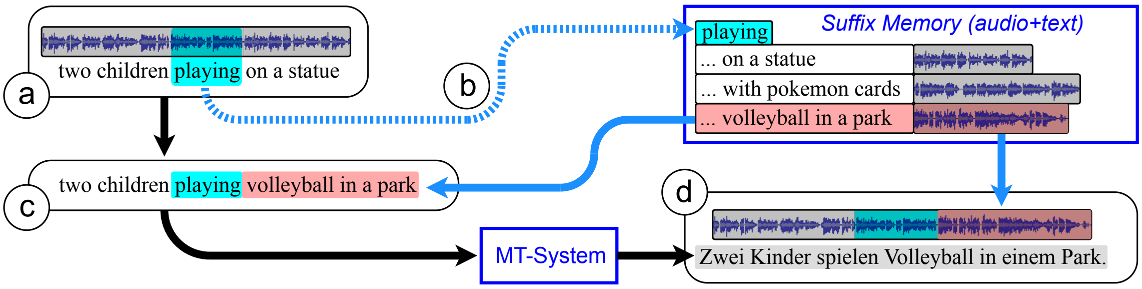

a
two children playing on a statue
b
Suffix Memory (audio+text)
playing
... on a statue
... with pokemon cards
... volleyball in a park
c
two children playing volleyball in a park
d
MT-System
Zwei Kinder spielen Volleyball in einem Park.

Figure 1: (a) Select a pivoting token, e.g., "playing". (b) Retrieve suitable text-audio entries from the suffix memory to sample a replacement. (c) Compile a new transcription containing prefix, pivoting token, and replacement suffix. (d) Recombine a new training example by translating the new transcription and concatenating the audio sections.

pairs and store the data in a key-value table. The audio entries contain only references to the original audio segments and our implementation is thus very memory efficient. We only utilize basic off-the-shelf components that are widely available and our suffix memory has a negligible memory footprint. Table 1 summarizes the number of additional training examples in each experiment.

Data	Baseline	KD	STR
CoVoST 2	288k	+288k	+255k
Europarl-ST (En-De)	3.25k	+3.25k	+2.78k
Europarl-ST (En-Fr)	3.17k	+3.17k	+2.71k

Table 1: Number of examples per configuration.

Model configuration All our implementations are based on FAIRSEQ (Wang et al., 2020; Ott et al., 2019).[4] In all speech-to-text tasks, we use the Transformer architecture (Vaswani et al., 2017) labelled as "s2t_transformer_s" in FAIRSEQ, which consists of convolutional layers for downsampling the input sequence with a factor of 4 before the self-attention layers. The encoder has 12 layers while the decoder has 6 layers with the dimensions of the self-attention layers set to 256 and the feed-forward network dimension set to 2048.

For the CoVoST 2 MT tasks, we use a smaller Transformer model of 3 layers for both encoder and decoder. The encoder-decoder embeddings and the output layer are shared. For the Europarl-ST MT tasks, we use the Transformer BASE configuration as described in Vaswani et al. (2017).

Training In the CoVoST 2 AST experiments, we use the character-level ASR model downloaded from the FAIRSEQ GitHub webpage[5] to initialize the encoder of the AST systems. Each AST system is then trained for another 50,000 steps. For Europarl-ST, we train a subword unit ASR system on the English audio-transcription pairs of the En-De data for 25,000 steps. The resulting ASR system is used to initialize both En-De and En-Fr AST systems which are trained for another 20,000 steps. Throughout all speech-to-text experiments, we apply gradient accumulation resulting in an effective mini-batch size of 160k frames. We use Adam optimizer (Kingma and Ba, 2015) with an inverse square root learning rate schedule. We use 10k steps for warmup and a peak learning rate of 2e-3. SpecAugment (Park et al., 2019) is applied with a frequency mask parameter of 27 and a time mask parameter of 100, both with 1 mask along their respective dimension. We perform validation and checkpoint saving after every 1,000 updates.

In case of the CoVoST 2 MT task, the Transformer model is pre-trained on in-domain data with 30,000 steps and an effective mini-batch size of 16,000 tokens. For the Europarl-ST dataset, the MT models are pre-trained on a combination of Europarl-ST and the additional training data. The Adam optimizer is used with an inverse square root learning rate schedule again, now with 4k steps for warmup and a peak learning rate of 5e-4. After pre-training, we finetune the model on the in-domain data with SGD and a constant learning rate of 5e-5.

Inference In the speech-to-text experiments, we average the 10 best checkpoints based on the validation loss. For the MT tasks, we average the 5 best checkpoints. Throughout all AST experiments and MT tasks, we apply beam search with a beam size of 5.

4 Results

Our experiments are focused on the improvements of our proposed method over KD alone on both CoVoST 2 and Europarl-ST datasets. We evaluate

[4] github.com/statnlp/str/, accessed 3/10/2022

[5] github.com/pytorch/fairseq/..., accessed 3/11/2022

model	En-De	En-Ca	En-Tr	En-Cy	En-Sl
Wang et al. (2021) Bi-AST	16.3	21.8	10.0	23.9	16.0
Baseline	17.22 ± 0.09	23.15 ± 0.10	10.31 ± 0.04	25.46 ± 0.08	15.64 ± 0.04
KD	18.26 ± 0.05	24.48 ± 0.16	11.10 ± 0.03	26.87 ± 0.16	17.21 ± 0.02
STR	18.77 ± 0.04	24.83 ± 0.12	11.62 ± 0.04	27.28 ± 0.11	17.54 ± 0.14
KD+STR	19.06 ± 0.02	25.33 ± 0.06	11.83 ± 0.01	27.73 ± 0.09	17.83 ± 0.09

Table 2: Averaged results in BLEU on the CoVoST 2 dataset over 3 runs with standard deviations (±). Models KD and KD+STR are significantly different for all language pairs with $p < 0.0002$ using a paired randomization test.

the translation results with both BLEU[6] (Papineni et al., 2002) and chrF2[7] (Popović, 2016) using the implementation of SACREBLEU (Post, 2018). Each experiment is repeated 3 times and we report mean and standard deviation.

We also conduct significance tests using a paired approximate randomization test (Riezler and Maxwell III, 2005) with default settings of SACREBLEU. We compute p-values between KD and KD+STR for each evaluated language pair of the experiments' datasets and between each run initialized with the same random seed. The individual p-values are reported in Appendix A.4.

Section 4.3 contains a discussion on the connection between STR- and MT-performance. We also report additional experiments which show how the amount of STR data affects the final performance in Section 4.4. An error analysis with examples and a discussion on the limitations of STR has been moved to Appendix A.5 due to space constraints.

4.1 Results on CoVoST 2

Table 2 lists BLEU scores on the five considered CoVoST 2 language pairs. Our baseline model is the AST system finetuned on the in-domain audio-translation pairs only. Its performance over the selected language pairs is quite diverse with BLEU scores ranging from 10.31 (En-Tr) to 25.46 (En-Cy). Our baseline models are comparable to and often better in terms of BLEU than the bilingual AST (Bi-AST) models by Wang et al. (2021).

Training together with translations generated by KD improves the baseline model by a substantial margin of 0.8 to 1.6 BLEU points. Our proposed STR method alone slightly surpasses the KD performance and brings further improvements when the augmented data is combined (KD+STR) with BLEU score increases ranging from 0.62 for En-Sl to 0.86 for En-Cy. In total, we observe BLEU score improvements of 1.5 to 2.3 for KD+STR. Since BLEU scores are often biased towards short translations, we additionally calculate chrF2 scores and report them in Appendix A.3.

We obtain significantly different models for all language pairs with $p < 0.0002$. This is strong evidence that the better performing models trained on KD+STR are different to the plain KD models.

4.2 Results on Europarl-ST

Table 3 lists the BLEU score results of Europarl-ST En-De and En-Fr. Similar to the results on CoVoST 2, the KD models bring substantial improvements over the baseline systems. The gains are 6.02 points for En-De and 6.27 points for En-Fr. We attribute this to the strong machine translation model that is trained on large amounts of additional training data (see Section 4.3 for more details on this). Our proposed STR method alone does not reach the KD performance but the combination KD+STR still delivers remarkable gains over KD, i.e., 1.13 points on En-De and 0.45 points on En-Fr, showing the complementarity of KD and STR. We also evaluate our models using chrF2. The numbers are listed in Appendix A.3.

model	En-De	En-Fr
Baseline	14.47 ± 0.16	22.52 ± 0.07
KD	20.49 ± 0.07	28.79 ± 0.14
STR	19.80 ± 0.14	28.01 ± 0.17
KD+STR	21.62 ± 0.12	29.28 ± 0.10

Table 3: Averaged results in BLEU on the Europarl-ST dataset over 3 runs with standard deviations (±). Models KD and KD+STR are significantly different for En-De with $p < 0.00025$. For En-Fr, we only found two runs to be significantly different with $p < 0.05$.

In the En-De experiments, we obtain significant differences between the KD and KD+STR models with $p < 0.00025$. For En-Fr, only two out of three runs show significant differences with $p < 0.05$. In terms of chrF2 scores, however, we found all compared models to be significantly different. See Appendix A.4 for details.

[6] nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.0.0

[7] nrefs:1|case:mixed|eff:yes|nc:6|nw:0|space:no|version:2.0.0

4.3 Connection to MT-Performance

To evaluate the dependency of STR on the MT-performance, we calculate BLEU scores for the MT-systems we use for CoVoST 2 and Europarl-ST data augmentation with STR and compare them in a cross-lingual manner. We see a noticeable correlation of MT-performance and STR-improvement.

On CoVoST 2, the highest improvement for STR is observed on the En-Cy language pair, which is also the best performing MT-model. The En-Ca language pair's MT-model also performs very well and shows the second highest gain for STR together with En-Sl. See Table 4 for more details.

On Europarl-ST, we observe a different behavior. While the MT-model for En-Fr is clearly better than the one for En-De, the gains are larger in the latter case. This might be due to the fact that the En-Fr ST-model already has a relatively high performance after training on KD alone (see Table 3). We also hypothesize that adding our STR method to KD is more useful if the sentence structure of source and target languages is very different. In case the alignments between source and target language are relatively parallel, KD already generates very useful examples and our approach can only introduce limited new information on top of that, e.g., by adding speaker variations. See Table 5 for the exact BLEU scores and improvements.

model	En-De	En-Ca	En-Tr	En-Cy	En-Sl
MT	30.05	39.66	21.28	43.57	30.32
STR-Δ	+1.84	+2.18	+1.51	+2.27	+2.19

Table 4: Machine translation performance measured in BLEU on the CoVoST 2 test set. The second row (STR-Δ) reports the BLEU improvements of KD+STR in comparison to the baseline.

model	En-De	En-Fr
MT	32.16	40.11
STR-Δ	+7.15	+6.76

Table 5: Machine translation performance measured in BLEU on the Europarl-ST test set. The second row (STR-Δ) reports BLEU improvements of KD+STR in comparison to the baseline.

4.4 Dependence on Amount of STR Data

We conduct an additional experiment on CoVoST 2 to evaluate the dependence of our STR method on the amount of generated training data. In Figure 2 we report the test performance on 5 language pairs of a single run (seed=0) after training on 1/3, 2/3, or all STR generated data. For some language pairs, we already observe large gains after using 1/3 or 2/3 of the total STR data. Most language pairs will further benefit from more additional data, while one language pair (En-Sl) seems to degrade when moving from 2/3 to all training data on this single run. Summarizing, we observe a trend on all but one language pair that more augmented data improves performance.

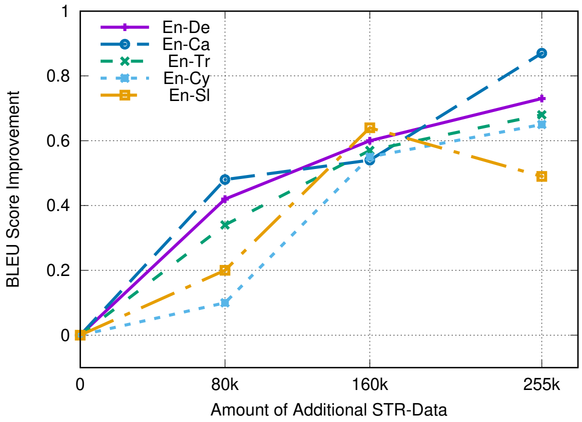


Figure 2: BLEU improvements for different amounts of STR augmented data on CoVoST 2 on a single run (seed=0) for 5 language pairs. We evaluate the addition of 0, 80k, 160k, and 255k STR-generated data points to the baseline KD data.

5 Conclusion

We proposed an effective data augmentation method for end-to-end speech translation which leverages audio alignments, linguistic properties, and translation. It creates new audio-translation pairs via *sampling* from a memory-efficient suffix memory, *translating* through an MT model and *recombining* original and sampled audio segments with translations. Our method achieves significant improvements over augmentation with KD alone on both large (CoVoST 2) and small scale (Europarl-ST) datasets. In future work, we would like to investigate the utility of other linguistic properties for AST augmentation and we would like to extend our method to multilingual AST.

Acknowledgements

This research was supported in part by the German research foundation DFG under grant RI-2221/4-1. We'd also like to thank the anonymous reviewers for their helpful comments.

References

Daniel Beck, Trevor Cohn, and Gholamreza Haffari. 2019. Neural speech translation using lattice transformations and graph networks. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 26–31, Hong Kong. Association for Computational Linguistics.

N. Bertoldi and M. Federico. 2005. A new decoder for spoken language translation based on confusion networks. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU 2005)*, pages 86–91. IEEE.

Hirofumi Inaguma, Tatsuya Kawahara, and Shinji Watanabe. 2021. Source and target bidirectional knowledge distillation for end-to-end speech translation. In *Proceedings of NAACL-HLT 2021*, pages 1872–1881, Online. Association for Computational Linguistics.

Javier Iranzo-Sánchez, Joan Albert Silvestre-Cerdà, Javier Jorge, Nahuel Roselló, Adrià Giménez, Albert Sanchis, Jorge Civera, and Alfons Juan. 2020. Europarl-st: A multilingual corpus for speech translation of parliamentary debates. In *Proceedings of ICASSP 2020*, pages 8229–8233, Barcelona, Spain. IEEE.

Ye Jia, Melvin Johnson, Wolfgang Macherey, Ron J. Weiss, Yuan Cao, Chung-Cheng Chiu, Naveen Ari, Stella Laurenzo, and Yonghui Wu. 2019. Leveraging weakly supervised data to improve end-to-end speech-to-text translation. In *Proceedings of ICASSP 2019*, Brighton, UK. IEEE.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR 2015*, San Diego, CA, USA.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of EMNLP 2018: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Tsz Kin Lam, Mayumi Ohta, Shigehiko Schamoni, and Stefan Riezler. 2021a. On-the-fly aligned data augmentation for sequence-to-sequence asr. In *Proceedings of INTERSPEECH 2021*, Brno, Czech Republic. ISCA.

Tsz Kin Lam, Shigehiko Schamoni, and Stefan Riezler. 2021b. Cascaded models with cyclic feedback for direct speech translation. In *Proceedings of ICASSP 2021*, pages 7508–7512, Toronto, ON, Canada. IEEE.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Proceedings of INTERSPEECH 2017*, volume 2017, pages 498–502, Stockholm, Sweden. ISCA.

Arya D. McCarthy, Liezl Puzon, and Juan Miguel Pino. 2020. Skinaugment: Auto-encoding speaker conversions for automatic speech translation. In *Proceedings of ICASSP 2020*, pages 7924–7928, Barcelona, Spain. IEEE.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, pages 48–53, Minneapolis, MN, USA. Association for Computational Linguistics.

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

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin Dogus Cubuk, and Quoc V. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. In *Proceedings of INTERSPEECH 2019*, pages 2613–2617, Graz, Austria. ISCA.

Juan Miguel Pino, Liezl Puzon, Jiatao Gu, Xutai Ma, Arya D. McCarthy, and Deepak Gopinath. 2019. Harnessing indirect training data for end-to-end automatic speech translation: Tricks of the trade. In *Proceedings of IWSLT 2019*, Hong Kong, China.

Maja Popović. 2016. chrF deconstructed: beta parameters and n-gram weights. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 499–504, Berlin, Germany. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Stefan Riezler and John T Maxwell III. 2005. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 57–64. Association for Computational Linguistics.

Nicholas Ruiz, Qin Gao, Will Lewis, and Marcello Federico. 2015. Adapting machine translation models toward misrecognized speech with text-to-speech pronunciation rules and acoustic confusability. In *Proceedings of INTERSPEECH 2015*, pages 2247–2251, Dresden, Germany. ISCA.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In *Proceedings of EACL 2021: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

Matthias Sperber and Matthias Paulik. 2020. Speech translation and the end-to-end promise: Taking stock of where we are. In *Proceedings of ACL 2020*, pages 7409–7421, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 2017*, pages 5998–6008, Long Beach, CA, USA.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020. fairseq s2t: Fast speech-to-text modeling with fairseq. In *Proceedings of AACL/IJCNLP 2020: System Demonstrations*, pages 33–39, Suzhou, China. Association for Computational Linguistics.

Changhan Wang, Anne Wu, Jiatao Gu, and Juan Pino. 2021. CoVoST 2 and Massively Multilingual Speech Translation. In *Proceedings of INTERSPEECH 2021*, pages 2247–2251, Brno, Czech Republic.

Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV 2019*, pages 6022–6031, Seoul, Korea (South). IEEE.

A Appendix

A.1 Data Description

CoVoST 2 is a large scale dataset of 430h English audio and 288k sentences for each language in the training set. The training set contains repetitions of the same sentence spoken by different speakers. We use the original data splits generated by the `get_covost_splits.py` script[8] on five languages pairs, namely English-German (En-De), English-Catalan (En-Ca), English-Turkish (En-Tr), English-Welsh (En-Cy) and English-Slovenian (En-Sl), resulting in about 15.5k sentences for each dev and test dataset.

Europarl-ST, in contrast, is a small AST dataset. It contains debates held in the European Parliament and their translations, thus representing a realistic AST scenario imposing very different challenges than the CoVoST 2 dataset. We conduct experiments on the English-German (En-De) and English-French (En-Fr) language pairs. The En-De data contains 89h of audio and 35.5k sentences. The En-Fr data contains 87h of audio and 34.5k sentences.

A.2 Filtering Criteria by the Acoustic Aligner

In very rare cases, the acoustic aligner does not return an alignment at all and we have to discard these examples. In some cases, the obtained alignments by the acoustic aligner are of low quality, i.e., contain alignments to unknown tokens. In such cases, if the number of tokens of the output transcriptions of the acoustic aligner matches the number of tokens in the input transcriptions, we can still use this alignment for data augmentation as alignments in ASR are always strictly parallel. Thus, if we cannot retrieve suitable alignments, we discard the example. This procedure reduces the amount of augmented data: we discard approximately 12% of the examples for CoVoST 2, and about 15% of the examples for Europarl-ST. See Table 1 for the final data sizes.

A.3 Additional chrF2 Scores

In this section, we additionally report chrF2 scores on CoVoST 2 and Europarl-ST datasets, since BLEU scores are often biased towards short translations. This issue is especially problematic on the CoVoST 2 datasets because of its large number of very short sentences. We list the CoVoST 2 chrF2 results in Table 10, and the Europarl-ST results in Table 6.

Our chrF2 results averaged over three runs confirm the improvements we observed throughout our experiments in terms of BLEU. When we look at chrF2, the better performing KD+STR models are always significantly different to the KD models. Even in case of the En-Fr language pair of the Europarl-ST dataset where we detected significant differences only in two of three runs in terms of BLEU, we found all three runs significantly different in terms of chrF2 with $p < 0.025$ this time. Detailed p-values per run are listed in Table 8 and 7 for our CoVoST 2 experiments, and in Table 9 for our Europarl-ST experiments.

A.4 Detailed p-values for System Comparison

Tables 7 and 8 report the exact p-values for comparison of KD and KD+STR models w.r.t. BLEU and chrF2 scores on CoVoST 2, respectively. Table 9 reports the exact p-values for comparison of KD and KD+STR models w.r.t. BLEU and chrF2 scores on Europarl-ST, respectively. We use the implementation of SACREBLEU for calculation.

[8] github.com/facebookresearch/covost, accessed 3/11/2022

model	En-De	En-Fr
Baseline	44.90 ± 0.22	48.60 ± 0.14
KD	51.43 ± 0.05	54.97 ± 0.05
STR	50.6 ± 0.0	54.1 ± 0.22
KD+STR	52.37 ± 0.09	55.37 ± 0.09

Table 6: Averaged results in chrF2 on En-De and En-Fr of Europarl-ST dataset over 3 runs with standard deviations (±). Models KD and KD+STR are significantly different for En-De with $p < 0.0002$ using a paired randomization test. For En-Fr, the models are significantly different with $p < 0.025$.

seed	En-De	En-Ca	En-Tr	En-Cy	En-Sl
0	0.0001	0.0001	0.0001	0.0001	0.0001
1	0.0001	0.0001	0.0001	0.0001	0.0001
321	0.0001	0.0001	0.0001	0.0001	0.0001

Table 7: The paired randomization test from SACREBLEU with default settings returned the following p-values for the three runs when comparing KD and KD+STR models' BLEU performance on CoVoST 2.

A.5 Examples and Error Analysis

We also take a look at the quality of our STR-augmented data and list examples in Table 11 and Table 12 for CoVoST 2 and Europarl-ST, respectively. Rows "src-A" and "src-B" contain the unmodified transcriptions from CoVoST 2 with our pivoting token underlined and segments we recombine in *italics*. The row "augm." shows the STR-augmented example, the row "transl." contains the MT-generated translation. The presented examples are the first 5 data examples taken directly from our augmented data set and are *not* cherry-picked.

Of the first five augmented examples from CoVoST 2 listed in Table 11, examples 1, 3, and 5 contain grammatically correct augmented source data (row "augm.") and the latter two are also semantically correct. Example 2 contains a grammatically wrong segment due to the problematic transcription of "src-B": here, the example is already an ungrammatical sentence and this transfers to our augmented example. Example 4 is also grammatically wrong. In this example, our augmentation method mixes the different senses of the word "directed" and produces a semantically incorrect result. This could be fixed by integrating more context, e.g., "directed through" can be used to disambiguate the different word senses of "directed".

Of the first five augmented examples from Europarl-ST in Table 12, examples 1, 3, and 5 are actually grammatically correct. Example 2 is grammatically wrong as our STR method does not respect the different grammatical forms of "pass" in "will pass" and "to pass", mixing up the two objects. Example 4 is also grammatically wrong, and it is again the wrong treatment of different grammatical forms of "do" in "do work" and "to do". These problems could be addressed by putting more effort into the suffix memory construction, e.g., by using n-grams as keys. Examples 3 and 5 demonstrate a property of Europarl-ST that partly explains the lower performance gain we observe for our STR-method here: there are many repetitive formalized sentences, and in these examples our augmentation method only differs by a single word from an already existing data example. Still, such augmented examples can be useful for training due to the speaker variations injected by STR.

We observe common errors in our augmented examples for CoVoST 2 and Europarl-ST that are often connected to the different word senses and syntactical functions of the selected pivoting token. However, even grammatically wrong sentences can sometimes be useful in training as they prevent overfitting on common structures in the data. Furthermore, the speaker variations in the examples that we produce can be helpful even if the augmented examples do not differ much from existing ones. Summarizing the error analysis, our simple STR-method is able to produce examples that are useful even with errors. Investigating more complex methods for better identification of pivoting tokens is a promising direction for future work.

seed	En-De	En-Ca	En-Tr	En-Cy	En-Sl
0	0.0001	0.0001	0.0001	0.0001	0.0001
1	0.0001	0.0001	0.0001	0.0001	0.0001
321	0.0001	0.0001	0.0001	0.0001	0.0001

Table 8: The paired randomization test from SACREBLEU with default settings returned the following p-values for the three runs when comparing KD and KD+STR models' chrF2 performance on CoVoST 2.

	BLEU		chrF2	
seed	En-De	En-Fr	En-De	En-Fr
0	0.0002	0.010	0.0001	0.016
1	0.0001	0.137	0.0001	0.021
321	0.0002	0.012	0.0001	0.005

Table 9: Conducting the paired randomization test from SACREBLEU with default settings returned the following p-values for the three runs when comparing KD and KD+STR models' performance on Europarl-ST.

model	En-De	En-Ca	En-Tr	En-Cy	En-Sl
Baseline	42.80 ± 0.08	46.63 ± 0.09	36.77 ± 0.09	49.13 ± 0.05	39.83 ± 0.05
KD	44.13 ± 0.05	48.17 ± 0.12	38.53 ± 0.05	50.67 ± 0.05	41.73 ± 0.05
STR	44.43 ± 0.05	48.60 ± 0.08	39.30 ± 0.08	51.03 ± 0.05	42.17 ± 0.05
KD+STR	45.13 ± 0.05	49.10 ± 0.08	39.70 ± 0.08	51.50 ± 0.00	42.60 ± 0.08

Table 10: Averaged results in chrF2 on the CoVoST 2 dataset over 3 runs with standard deviations (±). Models KD and KD+STR are significantly different for all language pairs with $p < 0.0002$ using a paired randomization test.

1	src-A	*these data components in turn* serve as the building blocks of data exchanges
	src-B	the governor appoints members of the board each of whom serve *seven years*
	augm.	*these data components in turn serve seven years*
	transl.	Diese Datenkomponenten wiederum servieren sieben Jahre.
2	src-A	*the church* is unrelated to the jewish political movement of zionism
	src-B	both sacks contain a man b is *on the left a on the right*
	augm.	*the church is on the left a on the right*
	transl.	Die Kirche befindet sich rechts auf der linken Seite.
3	src-A	*the following* represents architectures which have been utilized at one point or another
	src-B	monism sees brahma as the ultimate reality while monotheism represents *the personal form brahman*
	augm.	*the following represents the personal form brahman*
	transl.	Die folgende Darstellung repräsentiert die persönliche Form Brahman.
4	src-A	*additionally the pulse output can be* directed through one of three resonator banks
	src-B	he directed *no fewer than thirty seven productions at stratford*
	augm.	*additionally the pulse output can be directed no fewer than thirty seven productions at stratford*
	transl.	Darüber hinaus kann der Pulsausgang nicht weniger als siebenunddreißig Produktionen in Stratford geleitet werden.
5	src-A	*the two* are robbed by a pickpocket who is losing in gambling
	src-B	there are *six large portraits displayed in the senate chamber*
	augm.	*the two are six large portraits displayed in the senate chamber*
	transl.	Die beiden sind sechs große Porträts, die in der Senatskammer ausgestellt sind.

Table 11: The first 5 augmented data examples from CoVoST 2 for the En-De language pair. "src-A" and "src-B" are the unmodified transcriptions from CoVoST 2 with our pivoting token underlined and segments we recombine in *italics*. The "augm." row shows the STR-augmented example. The "transl." row contains the MT-generated translation.

1	src-A	*i would just like to say that there are more amendments in my report because my committee* has been more ambitious in the improvements it wanted to make to the commission proposal
	src-B	economic cooperation has *always been europe s most powerful engine for greater integration and europe has owed its success to this pragmatic approach since 1956*
	augm.	*i would just like to say that there are more amendments in my report because my committee* has *always been europe s most powerful engine for greater integration and europe has owed its success to this pragmatic approach since 1956*
	transl.	Je voudrais juste dire qu ’ il y a plus de modifications dans mon rapport, car ma commission a toujours été le moteur le plus puissant de l ’ Europe pour une plus grande intégration, et l ’ Europe doit son succès à cette approche pragmatique depuis 1956.
2	src-A	*i would like to thank all my colleagues on the committee who worked with me to put together some really big compromise amendments which we will* pass today
	src-B	the right of every member state to pass *laws as it deems fit as long as it has a democratic majority and that those laws should be recognised by other countries*
	augm.	*i would like to thank all my colleagues on the committee who worked with me to put together some really big compromise amendments which we will* pass *laws as it deems fit as long as it has a democratic majority and that those laws should be recognised by other countries*
	transl.	Je tiens à remercier tous mes collègues de la commission qui ont travaillé avec moi pour mettre en place des amendements de compromis vraiment importants, que nous adopterons des lois, tant qu ’ elle a une majorité démocratique et que ces lois devraient être reconnues par d ’ autres pays.
3	src-A	*i would* like all of you to give us a huge majority for this so that when we come to negotiate with the commission and council we will do our very best for europe s consumers
	src-B	i would also like *to thank all the shadow rapporteurs*
	augm.	*i would* like *to thank all the shadow rapporteurs*
	transl.	Je tiens à remercier tous les rapporteurs fictifs.
4	src-A	*mr president let us hope that the american proposals for purchases of toxic assets* do work because if they do not the contagion will almost certainly spread over here
	src-B	what we really need to do *is empower women*
	augm.	*mr president let us hope that the american proposals for purchases of toxic assets* do *is empower women*
	transl.	Monsieur le Président, espérons que les propositions américaines d ’ achats d ’ actifs toxiques permettent aux femmes.
5	src-A	*i would* like assurance from mr jouyet and mr almunia that we really do have our defences in place
	src-B	mr president i would like *to thank the rapporteurs and other shadows for the hard work they have put into producing these reports*
	augm.	*i would* like *to thank the rapporteurs and other shadows for the hard work they have put into producing these reports*
	transl.	Je voudrais remercier les rapporteurs et d ’ autres ombres pour le travail qu ’ ils ont accompli dans la production de ces rapports.

Table 12: The first 5 augmented data examples from Europarl-ST for the En-Fr language pair. “src-A” and “src-B” are the unmodified transcriptions from Europarl-ST with our pivoting token underlined and segments we recombine in *italics*. The “augm.” row shows the STR-augmented example. The “transl.” row contains the MT-generated translation.

Predicting Sentence Deletions for Text Simplification Using a Functional Discourse Structure

Bohan Zhang*
University of Michigan
zbohan@umich.edu

Prafulla Kumar Choubey
Salesforce Research
pchoubey@salesforce.com

Ruihong Huang
Texas A&M University
huangrh@tamu.edu

Abstract

Document-level text simplification often deletes some sentences besides performing lexical, grammatical or structural simplification to reduce text complexity. In this work, we focus on sentence deletions for text simplification and use a news genre-specific functional discourse structure, which categorizes sentences based on their contents and their function roles in telling a news story, for predicting sentence deletion. We incorporate sentence categories into a neural net model in two ways for predicting sentence deletions, either as additional features or by jointly predicting sentence deletions and sentence categories. Experimental results using human-annotated data show that incorporating the functional structure improves the recall of sentence deletion prediction by 6.5% and 10.7% respectively using the two methods, and improves the overall F1-score by 3.6% and 4.3% respectively.

1 Introduction

Text simplification aims to rewrite complex texts in order to make them easier to read and understand. This task can benefit vast low literacy readers, including children, language learners and people with aphasia, and has recently attracted increasing attention from the research community (Xu et al., 2016; Zhao et al., 2018; Martin et al., 2019; Dong et al., 2019). However, most previous research has focused on sentence-level text simplification and aim to simplify one sentence at a time. As a result, few discourse-level phenomena have been examined or understood for achieving document-level text simplification.

Sentence deletion is a commonly used strategy to achieve intense simplification (Drndarevic and Saggion, 2012; Woodsend and Lapata, 2011), i.e., some less important sentences from an original article are simply deleted and ignored for simplification. While professional re-writers may consider many factors and use several measures of *importance* to decide if a sentence should be deleted, some discourse structures provide automated measures to derive *importance* for sentences in a document. In particular, functional discourse structures categorize text units (sentences or paragraphs) based on their contents and their function roles in serving the purpose of a specific text-genre, such as scientific papers (Teufel et al., 1999; Liakata et al., 2012) and news articles (Yarlott et al., 2018; Choubey et al., 2020), and are therefore, expected to directly reveal the *importance* of a sentence within a document.

In this work, we explore the use of news genre-specific functional structures for predicting sentence deletions in news documents. Specifically, we use news discourse profiling structure, which categorizes contents of news articles around the main news event, constructed through a publicly available system (Choubey et al., 2020)[1]. This system labels each sentence with one of eight content types reflecting common discourse roles of a sentence in telling a news story, including two content types for sentences describing the main news event and its immediate consequences (*main content*), two content types for sentences providing *context-informing contents* and four content types for sentences providing *further supportive information* in a news article.

We perform experiments using the Newsela corpus (Xu et al., 2015), a widely used dataset for text simplification research that contains 1492 English news articles and four simplified versions for each news article targeting audience of different reading levels (from elementary to high school students). Since we aim to achieve maximal simplification, we predict sentence deletions for tar-

*Most work was done while Bohan was a summer intern in the NLP lab at Texas A&M University.

[1] This system can be found here: https://github.com/prafulla77/Discourse_Profiling.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 255 - 261
May 22-27, 2022 ©2022 Association for Computational Linguistics

get reading level corresponding to the elementary school students. We first build a document-level neural network as the basic model for predicting sentence deletions. We then incorporate content types of sentences into the prediction system using two methods, 1) by using content type labels as additional features to enrich sentence representations, and 2) by jointly predicting both sentence deletion labels and discourse content type labels. Experimental results show that, with little to no drop on precision, both methods for incorporating sentence content type information improve the recall (F1 score) on the sentence deletion prediction task by 6.5% (3.6%) and 10.7% (4.3%) respectively. Analysis on the development set shows that the additional deletions correctly recognized by our system are all sentences providing context-informing or supportive contents.

2 Related Work

The previous research on text simplification has focused on word or phrase level simplification (Yatskar et al., 2010; Biran et al., 2011; Specia et al., 2012; Paetzold and Specia, 2017), or sentence-level simplification (Wubben et al., 2012; Sutskever et al., 2014; Nisioi et al., 2017; Zhao et al., 2018; Dong et al., 2019), few research has been conducted for document-level text simplification.

Sentence deletion, as an interesting phenomenon for document-level text simplification, has been studied in several pilot studies. (Petersen and Ostendorf, 2007) conducted a corpus analysis and showed that sentence position and content influence sentence deletion or retention. The recent pilot research for sentence deletion prediction (Zhong et al., 2019) considers sentence position in a document, document length and topic, as well as exploits rhetorical discourse structures that capture text coherence in general and can be used to derive the *salience* of a sentence in a discourse. However, while sentence position and the two document characteristics are shown useful for sentence deletion prediction, discourse features based on rhetorical discourse structures are shown to have little impact for this task. Compared to general rhetorical discourse structures that do not consider genre specialties, the genre-specific functional structure we examine in this paper can more directly reveal the importance of a sentence within a document.

3 The News Discourse Structure and Sentence Types

News discourse profiling (Choubey et al., 2020) categorizes sentences in news articles into eight schematic categories that describe the common discourse roles of sentences in telling a news story, following the news content schemata proposed by Van Dijk (Teun A, 1986; Van Dijk, 1988a,b). These eight sentence categories fall into three groups.

Main Contents: are the most relevant information of news articles, including sentences that introduce the main event as the major subjects of a news article (**Main Event**), and sentences that describe consequence events immediately triggered by the main event (**Consequence**).

Context Informing Contents: provide information of the actual situation in which main event occurred, including sentences that describe the recent events that act as possible causes or preconditions for the main event (**Previous Events**), and sentences that describe ongoing situation and other context informing contents (**Current Context**).

Additional Supportive Contents: contain the least relevant information, including sentences that describe past events that precede the main events in months and years (**Historical Event**), sentences that describe unverifiable situations, fictional or personal account of incidents of an unknown person (**Anecdotal Event**), opinionated contents that describe reactions from immediate participants, experts, known personalities as well as journalist or news source (**Evaluation**), and speculations on the possible consequences of the main or contextual events (**Expectation**)

3.1 Analysis of Deletions w.r.t Sentence Types

We conducted an analysis on deletion rate for each sentence category using the development set (Section 5.1) which was manually annotated with sentence deletion labels. The discourse content type labels of sentences were predicted by the news discourse profiling system (Choubey et al., 2020). Table 1 shows the results. We can see that **Main Event** sentences have the lowest deletion rate of 14.7%, much lower than other types of sentences. **Previous Event** sentences, as one type of context informing contents, have a relatively low deletion rate as well to provide necessary context, i.e., possible causes or preconditions, to understand the main news events. While additional supportive contents overall have a high deletion rate, **Anecdotal**

The NFL delivered that message in a resounding way Monday, suspending the New England Patriots star without pay for the first four games of next season for "conduct detrimental to the integrity of the NFL." (Main Event)

The punishment comes days after the league announced results of an investigation that found Brady was "likely generally aware" that equipment assistants employed by the team had conspired to deflate the Patriots' footballs for last season's AFC championship game, making the balls easier to throw and catch. (Previous Event)

The Patriots also were fined $1 million — equaling the largest in league history — and stripped of their first-round draft pick next year and a fourth-round selection in 2017. (Consequence)

The Pariots have been accused of cheating in the past, and in 2007 were caught breaking league rules by videotaping the sideline hand signals of New York Jets coaches. (Historical Event)

That incident, nicknamed Spygate, cost New England coach Bill Belichick $500,000 and the league docked the Patriots a first-round draft pick. (Historical Event)

By every indication, the incident has not dimmed Brady's star one iota among Patriots' fans. (Current Context)

He was cheered enthusiastically last week, one day after the Wells report was released, when he spoke at an event at Salem State University in Massachusetts. (Current Context)

Figure 1: An example article: Brady Deflategate. Sentences in purple were deleted for text simplification.

	Main Event	Consequence	Previous Event	Current Context	Historical Event	Anecdotal Event	Evaluation	Expectation
Deleted	5 (14.7)	0 (NA)	7 (31.8)	128 (37.5)	36 (46.2)	11 (27.5)	206 (41.2)	35 (33.7)
Retained	29 (85.3)	0 (NA)	15 (68.2)	213 (62.5)	42 (53.8)	29 (72.5)	294 (58.8)	69 (66.3)

Table 1: The number (percentage) of sentences in each type that are deleted or retained, on the development set. The news discourse profiling system did not label any sentence in the development set as Consequence, which is a minority class as revealed by (Choubey et al., 2020)

Event sentences have a low deletion rate, possibly because personal account of incidents present especially interesting contents for elementary students, the target group of our chosen simplification level.

Figure 1 shows an example document where both deleted sentences (colored in purple) are of one additional supportive content type, **Historical Event**.

4 Models

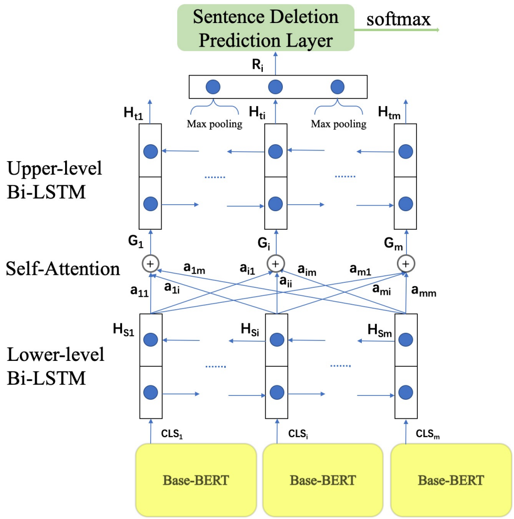


Figure 2: The Baseline Model

As a baseline model, (shown in Figure 2), we built a document-level neural network model to learn context aware sentence representations for predicting sentence deletions. Similar architectures have been shown useful for several other discourse-level tasks (Nallapati et al., 2016; Choubey et al., 2020).

Specifically, the model takes a document as input and has two document-level BiLSTM layers (Hochreiter and Schmidhuber, 1997) stacked up with a self-attention layer between them, to sufficiently exploit document wide contexts for building sentence representations. In addition, for each sentence, we further concatenate its sentence representation with two vectors obtained by max pooling over representations of its surrounding sentences (two sentences to each side), to obtain the final sentence representation R_i, that is better aware of the local context. We use a feed forward neural network with 1024-2 units to predict a binary label (deleted or not) for each sentence[2] based on its final sentence representation. We apply base BERT (Devlin et al., 2019) to obtain the initial sentence representations of 768 dimensions. Both BiLSTMs

[2] We also tried to add a CRF layer to capture deletion label dependencies between sentences, and predict labels for a sequence of sentences in a document, however, it did not improve the sentence deletion prediction performance.

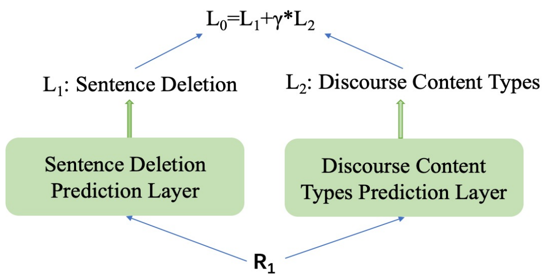

L0=L1+γ*L2
L1: Sentence Deletion
L2: Discourse Content Types
Sentence Deletion Prediction Layer
Discourse Content Types Prediction Layer
R1

Figure 3: Jointly Predicting Two Types of Labels

have the hidden dimension size of 512.

Next, we present two methods to utilize the functional structure for sentence deletion prediction.

4.1 Feature Concatenation

For each sentence, we create a feature vector F_i with eight dimensions corresponding to the eight discourse content types[3], and values in the vector are probabilities of content types for the target sentence as output by the news discourse profiling system. We concatenate the feature vector F_i with the final sentence representation R_i and feed the concatenated vector to the sentence deletion prediction layer.

4.2 Joint Learning

Instead of creating features, we learn to jointly predict both sentence deletion labels and discourse content type labels (system predicted) using shared sentence representations (Figure 3). Specifically, we add a new prediction layer with 1024-9[4] units to predict discourse content types for sentences, and learn to jointly predict both types of labels by minimizing the aggregated loss of two tasks: $L_0 = L_1 + \gamma * L_2$, where L_1 is the cross-entropy loss for the sentence deletion prediction task and L_2 is the mean squared loss for the discourse content type prediction task[5].

5 Evaluation

5.1 Dataset

We conduct experiments using the Newsela corpus for text simplification (Xu et al., 2015). This corpus contains 1492 English news articles and four simplified versions for each article targeting students ranging from grade 2 to grade 12. In our study, we focus on predicting sentence deletions to achieve the relatively aggressive level of simplification that targets elementary school students (grades 2 to 5).

Test and Development Data: We created a new annotated dataset. The annotated dataset of 50 documents used in Zhong et al. (2019) was not released yet when we started to work on this project. Our code and the method to obtain our annotated dataset can be found on github[6].

Different from the crowd-sourcing based annotation method of Zhong et al. (2019) that decomposes the document-level sentence alignment task to a paragraph alignment task followed by a paragraph-level sentence alignment task, we ask our two annotators to read through a whole news article and its simplified article before annotating alignment sentence by sentence, which enables thorough annotations. Then, for each sentence in an original article, we instruct our annotators to align it with all the sentences in the simplified article that contain part or all of its contents (or paraphrases), one sentence in an original article will be labeled as "deleted" if *no* sentence in its simplified article is aligned with this sentence.

We annotated 95 (containing 4,334 sentences) randomly selected news articles. The two annotators first annotated five news articles (228 sentences) in common and achieved a high kappa agreement (Artstein and Poesio, 2008) of 0.911. Then, each of them annotated 45 more articles. We randomly selected 25 annotated articles and use them as the development set, and use the other 70 articles as the test set. 48% and 38% of sentences are annotated as deleted in the test and development sets respectively. We will publish our annotations.

Training Data: We create noisy supervision to train the systems by applying an automatic sentence alignment tool CATS[7] (Štajner et al., 2018) to the remaining 1397 unlabeled news articles and quickly obtained alignments between these news

[3] Document length and sentence position in a document have been shown useful for sentence deletion prediction in the previous work when used in a feature based approach (Zhong et al., 2019). We also concatenated these features with the final sentence representations. However, these features hurt the performance a little in our system, so we removed them. We suspect that document length and sentence position have been captured by the document-level neural net model and adding the features cause redundancies.

[4] Eight discourse content types plus one "Other" category.

[5] The mean squared loss is calculated against probabilities of content types for the target sentence as output by the news discourse profiling system.

[6] https://github.com/XMUBQ/SentenceDeletion

[7] CATS is a lexical similarity based sentence/paragraph alignment tool specifically designed for text simplification, and has been shown to perform well on the Newsela corpus.

	Current Context	Historical Event	Anecdotal Event	Evaluation	Expectation
Feature Concatenation	24	7	6	20	3
Joint Learning	20	3	3	21	1

Table 2: Numbers of additional deleted sentences from each content type that were correctly predicted. None of the correctly deleted sentences are from main event, consequence, and previous event content types.

Models	Dev Set	Test Set
FNN (Zhong et al., 2019)	44.6/60.4/51.3	56.7/57.2/57.0
Our Baseline	52.0/62.2/56.6	63.4/60.8/62.0
Feature Concatenation	**52.7**/64.8/58.1	**64.0**/67.3/65.6
Joint Learning	50.7/**69.8**/**58.7**	61.8/**71.5**/**66.3**

Table 3: Sentence deletion prediction results (P/R/F) (our dataset). Statistical significance tests show that compared with our baseline, both methods achieved significant improvements (p<0.01) in F1 measure.

Models	Dev Set	Test Set
FNN (Zhong et al., 2019)	61.7/60.7/61.0	56.8/60.6/58.6
Our Baseline	63.8/67.2/65.4	59.2/63.3/61.2
Feature Concatenation	69.7/**70.2**/70.0	**61.8**/66.1/63.9
Joint Learning	**70.9**/69.8/**70.4**	59.9/**68.6**/**63.9**

Table 4: Sentence deletion prediction results (P/R/F) (on the dataset from Zhong et al. (2019). Note that the results are not directly comparable with those in Zhong et al. (2019), as the training datasets are different. We used the Newsela corpus of a newer version and different automatic alignment tools to build our training dataset.

articles and their simplified articles. 82.11% of sentence alignments produced by CATS are correct when evaluated on our development set.

5.2 Experimental Settings

For regularization, we use dropout of 0.5 on the output of both BiLSTMs and the self-attention layer. We apply Adam optimizer (Kingma and Ba, 2014) for training, and the learning rate is set to 3e-4. All the neural models are trained for 15 epochs and we use the epoch yielding the best validation performance. We searched the hyper-parameter γ value over the range [0, 3] with a step size of 0.5, and its best value equals to 1.5.

5.3 Results and Analysis

In Table 3, we report the performance of our baseline and the two news discourse profiling structure-aware models. For better positioning of our work, we also re-implemented the model proposed in a recent work by Zhong et al. (2019), a feedforward neural network (FNN) model with sparse features[8]. First, our baseline system performs better than the feature based FNN model with 5.3% and 5.0% higher F1 score on validation and test datasets respectively. Then, both methods for incorporating discourse information have noticeably improved the performance on sentence deletion prediction. We also evaluate the models on the dataset from Zhong et al. (2019). As shown in Table 4, similar trends were observed on this dataset as well.

Since the performance gains of both discourse-aware models are mainly on recall, we analyze the distribution of additional deleted sentences correctly predicted by the two models. As shown in Table 2, the additional deleted sentences are either context informing contents or additional supportive contents, but none is main content. This observation corroborates our analysis in section 3.1.

6 Conclusion

We study sentence deletion prediction to achieve document-level text simplification. We have showed that a genre-specific functional discourse structure improves the prediction performance by large margins, when incorporated into a neural net model either as new features or for joint learning. For future work, we will study other useful discourse-level factors for sentence deletion prediction, we will also investigate multi-task learning to benefit both sentence deletion prediction and discourse parsing tasks.

7 Acknowledgements

We gratefully acknowledge support from National Science Foundation via the awards IIS-1942918. We would also like to thank the anonymous reviewers for their feedback.

[8] Some features have little impact to the performance, we only implemented the ones that have been shown useful in their ablation study, specifically, the document length and sentence position features. The model parameters and training settings were identical to the paper.

References

Ron Artstein and Massimo Poesio. 2008. Survey article: Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Or Biran, Samuel Brody, and Noémie Elhadad. 2011. Putting it simply: a context-aware approach to lexical simplification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 496–501, Portland, Oregon, USA. Association for Computational Linguistics.

Prafulla Kumar Choubey, Aaron Lee, Ruihong Huang, and Lu Wang. 2020. Discourse as a function of event: Profiling discourse structure in news articles around the main event. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5374–5386, Online. Association for Computational Linguistics.

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

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

Biljana Drndarevic and Horacio Saggion. 2012. Reducing text complexity through automatic lexical simplification: An empirical study for spanish. *Procesamiento del lenguaje natural*, 49:13–20.

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

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Maria Liakata, Shyamasree Saha, Simon Dobnik, Colin Batchelor, and Dietrich Rebholz-Schuhmann. 2012. Automatic recognition of conceptualization zones in scientific articles and two life science applications. *Bioinformatics*, 28(7):991–1000.

Louis Martin, Benoît Sagot, Éric de la Clergerie, and Antoine Bordes. 2019. Controllable sentence simplification. *arXiv preprint arXiv:1910.02677*.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P. Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 85–91, Vancouver, Canada. Association for Computational Linguistics.

Gustavo Paetzold and Lucia Specia. 2017. Lexical simplification with neural ranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 34–40, Valencia, Spain. Association for Computational Linguistics.

Sarah E Petersen and Mari Ostendorf. 2007. Text simplification for language learners: a corpus analysis. In *Workshop on Speech and Language Technology in Education*.

Lucia Specia, Sujay Kumar Jauhar, and Rada Mihalcea. 2012. SemEval-2012 task 1: English lexical simplification. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 347–355, Montréal, Canada. Association for Computational Linguistics.

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

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Simone Teufel, Jean Carletta, and Marc Moens. 1999. An annotation scheme for discourse-level argumentation in research articles. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, Bergen, Norway. Association for Computational Linguistics.

Van Dijk Teun A. 1986. News schemata. *Studying writing: linguistic approaches*, 1:155–186.

Teun A Van Dijk. 1988a. News analysis. *Case Studies of International and National News in the Press. New Jersey: Lawrence*.

Teun A Van Dijk. 1988b. News as discourse. Hillsdale,NJ, US: Lawrence Erlbaum Associates, Inc.

Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the*

2011 Conference on Empirical Methods in Natural Language Processing, pages 409–420, Edinburgh, Scotland, UK. Association for Computational Linguistics.

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

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

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

W. Victor Yarlott, Cristina Cornelio, Tian Gao, and Mark Finlayson. 2018. Identifying the discourse function of news article paragraphs. In *Proceedings of the Workshop Events and Stories in the News 2018*, pages 25–33, Santa Fe, New Mexico, U.S.A. Association for Computational Linguistics.

Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 365–368, Los Angeles, California. Association for Computational Linguistics.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. 2018. Integrating transformer and paraphrase rules for sentence simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3164–3173, Brussels, Belgium. Association for Computational Linguistics.

Yang Zhong, Chao Jiang, Wei Xu, and Junyi Jessy Li. 2019. Discourse level factors for sentence deletion in text simplification. *arXiv preprint arXiv:1911.10384*.

Multilingual Pre-training with Language and Task Adaptation for Multilingual Text Style Transfer

Huiyuan Lai, Antonio Toral, Malvina Nissim
CLCG, University of Groningen / The Netherlands
{h.lai, a.toral.ruiz, m.nissim}@rug.nl

Abstract

We exploit the pre-trained seq2seq model mBART for multilingual text style transfer. Using machine translated data as well as gold aligned English sentences yields state-of-the-art results in the three target languages we consider. Besides, in view of the general scarcity of parallel data, we propose a modular approach for multilingual formality transfer, which consists of two training strategies that target adaptation to both language and task. Our approach achieves competitive performance without monolingual task-specific parallel data and can be applied to other style transfer tasks as well as to other languages.

1 Introduction

Text style transfer (TST) is a text generation task where a given sentence must get rewritten changing its style while preserving its meaning. Traditionally, tasks such as swapping the polarity of a sentence (e.g. “This restaurant is getting worse and worse.”↔“This restaurant is getting better and better.”) as well as changing the formality of a text (e.g. “it all depends on when ur ready.”↔“It all depends on when you are ready.”) are considered as instances of TST. We focus here on the latter case only, i.e. *formality transfer*, because (i) recent work has shown that polarity swap is less of a style transfer task, since meaning is altered in the transformation (Lai et al., 2021a), and (ii) data in multiple languages has recently become available for formality transfer (Briakou et al., 2021b).

Indeed, mostly due to the availability of parallel training and evaluation data, almost all prior TST work focuses on monolingual (English) text (Rao and Tetreault, 2018; Li et al., 2018; Prabhumoye et al., 2018; Cao et al., 2020).[1] As a first step towards multilingual style transfer, Briakou et al. (2021b) have released XFORMAL, a benchmark of multiple formal reformulations of informal text in Brazilian Portuguese (BR-PT), French (FR), and Italian (IT). For these languages the authors have manually created evaluation datasets. On these, they test several monolingual TST baseline models developed using language-specific pairs obtained by machine translating GYAFC, a English corpus for formality transfer (Rao and Tetreault, 2018). Briakou et al. (2021b) find that the models trained on translated parallel data do not outperform a simple rule-based system based on handcrafted transformations, especially on content preservation, and conclude that formality transfer on languages other than English is particularly challenging.

One reason for the poor performance could be the low quality (observed upon our own manual inspection) of the pseudo-parallel data, especially the informal side. Since machine translation systems are usually trained with formal texts like news (Zhang et al., 2020), informal texts are harder to translate, or might end up more formal when translated. But most importantly, the neural models developed by Briakou et al. (2021b) do not take advantage of two recent findings: (i) pre-trained models, especially the sequence-to-sequence model BART (Lewis et al., 2020), have proved to help substantially with content preservation in style transfer (Lai et al., 2021b); (ii) Multilingual Neural Machine Translation (Johnson et al., 2017; Aharoni et al., 2019; Liu et al., 2020) and Multilingual Text Summarization (Hasan et al., 2021) have achieved impressive results leveraging multilingual models which allow for cross-lingual knowledge transfer.

In this work we use the multilingual large model mBART (Liu et al., 2020) to model style transfer in a multilingual fashion exploiting available parallel data of one language (English) to transfer the task and domain knowledge to other target languages. To address real-occurring situations, in our experiments we also simulate complete lack of parallel data for a target language (even machine translated),

[1] “Parallel data” in this paper refers to sentence pairs in the same language, with the same content but different formality.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 262 - 271
May 22-27, 2022 ©2022 Association for Computational Linguistics

and lack of style-related data at all (though availability of out-of-domain data). Language specificities are addressed through adapter-based strategies (Pfeiffer et al., 2020; Üstün et al., 2020, 2021). We obtain state-of-the-art results in all three target languages, and propose a modular methodology that can be applied to other style transfer tasks as well as to other languages. We release our code and hopefully foster the research progress.[2]

2 Approach and Data

As a base experiment aimed at exploring the contribution of mBART (Liu et al., 2020; Tang et al., 2020) for multilingual style transfer, we fine-tune this model with parallel data specifically developed for style transfer in English (original) and three other languages (machine translated).

Next, in view of the common situation where parallel data for a target language is not available, we propose a two-step adaptation training approach on mBART that enables modular multilingual TST. We avoid iterative back-translation (IBT) (Hoang et al., 2018), often used in previous TST work (Prabhumoye et al., 2018; Lample et al., 2019; Yi et al., 2020; Lai et al., 2021a), since it has been shown to be computationally costly (Üstün et al., 2021; Stickland et al., 2021a). We still run comparison models that use it.

In the first adaptation step, we address the problem of some languages being not well represented in mBART, which preliminary experiments have shown to hurt our downstream task.[3] We conduct a language adaptation denoising training using unlabelled data for the target language. In the second step, we address the task at hand through fine-tuning cross-attention with auxiliary gold parallel English data adapting the model to the TST task.

For TST fine-tuning, we use parallel training data, namely formal/informal aligned sentences (both manually produced for English and machine translated for three other languages). For the adaptation strategies, we also collect formality and generic non-parallel data. Details follow.

English formality data GYAFC (Rao and Tetreault, 2018) is an English dataset of aligned formal and informal sentences. Gold parallel pairs are provided for training, validation, and test.

Multilingual formality data XFORMAL (Briakou et al., 2021b) is a benchmark for multilingual formality transfer, which provides an evaluation set that consists of four formal rewrites of informal sentences in BR-PT, FR, and IT. This dataset contains pseudo-parallel corpora in each language, obtained via machine translating the English GYAFC pairs.

Language-specific formality non-parallel data Following Rao and Tetreault (2018) and Briakou et al. (2021b), we crawl the domain data in target language from Yahoo Answers.[4] We then use the style regressor from Briakou et al. (2021a) to predict formality score σ of the sentence to automatically select sentences in each style direction.[5]

Language-specific generic non-parallel data 5 M sentences containing 5 to 30 words for each language randomly selected from News Crawl.[6]

3 Adaptation Training

To adapt mBART to multilingual TST, we employ two adaptation training strategies that target language and task respectively.

3.1 Language Adaptation

As shown in Figure 1(a), we introduce a module for language adaptation. Inspired by previous work (Houlsby et al., 2019; Bapna and Firat, 2019), we use an adapter (ADAPT; ~50M parameters), which is inserted into each layer of the Transformer encoder and decoder, after the feed-forward block.

Following Bapna and Firat (2019), the ADAPT module A_i at layer i consists of a layer-normalization LN of the input $x_i \in \mathbb{R}^h$ followed by a down-projection $W_{down} \in \mathbb{R}^{h\times h}$, a non-linearity and an up-projection $W_{up} \in \mathbb{R}^{h\times h}$ combined with a residual connection with the input x_i:

$$A(x_i) = W_{up}\text{RELU}(W_{down}\text{LN}(x_i)) + x_i \quad (1)$$

Language adaptation training Following mBART's pretraining, we conduct the language adaptation training on a denoising task, which aims to reconstruct text from a corrupted version:

$$L_{\phi_{\boldsymbol{A}}} = -\sum \log(T \mid g(T); \phi_{\boldsymbol{A}}) \quad (2)$$

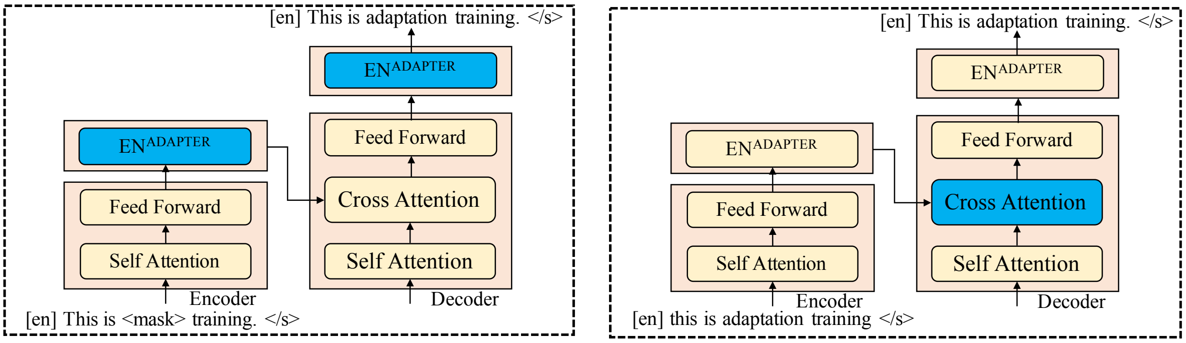
[2] All code at https://github.com/laihuiyuan/multilingual-tst.

[3] The number of monolingual sentences used in mBART-50's pre-training is only 49,446 for Portuguese, for example, versus 36,797,950 for French and 226,457 for Italian.

[4] https://webscope.sandbox.yahoo.com/catalog.php?datatype=l&did=11

[5] Sentences with $\sigma < -0.5$ are considered informal while > 1.0 are formal in our experiments.

[6] http://data.statmt.org/news-crawl/



(a) Language adaptation training with monolingual data

(b) Task adaptation training with English parallel data

Figure 1: Overview of adaptation training. In 1(a), the feed-forward network of each transformer layer or the inserted adapter layer is trained with monolingual data to adapt to the target language. In 1(b), the cross-attention of mBART is trained with auxiliary English parallel data to adapt to the TST task.

where $\phi_{\boldsymbol{A}}$ are the parameters of adaptation module A, T is a sentence in target language and g is the noise function that masks 30% of the words in the sentence. Each language has its own separate adaptation module. During language adaptation training, the parameters of the adaptation module are updated while the other parameters stay frozen.

3.2 Task Adaptation

As shown in Figure 1(b), after training the language adaptation module we fine-tune the model on the auxiliary English parallel data with the aim of making the model adapt to the specific task of formality transfer. Following Stickland et al. (2021b), we only update the parameters of the decoder's cross-attention (i.e. task adaptation module) while the other parameters are fixed, thus limiting computational cost and catastrophic forgetting.

Multilingual TST process For the language adaptation modules we have two settings: (i) adaptation modules $\mathbf{A}_s^E$ on the encoder come from the model trained with source style texts, and modules $\mathbf{A}_t^D$ on the decoder come from the model trained with target style texts (M2.X, Table 1); (ii) both $\mathbf{A}^E$ and $\mathbf{A}^D$ are from a model trained with generic texts (M3.X), so there are no source and target styles for the adaptation modules. For the task adaptation modules, we also have two settings: (i) the module is from the English model (X + EN cross-attn); (ii) fine-tuning the model of the target language with English parallel data (X + EN data).

4 Experiments

All experiments are implemented atop Transformers (Wolf et al., 2020) using mBART-large-50 (Tang et al., 2020). We train the model using the Adam optimiser (Kingma and Ba, 2015) with learning rate 1e-5 for all experiments. We train the language adaptation modules with generic texts separately for each language for 200k training steps with batch size 32, accumulating gradients over 8 update steps, and set it to 1 for other training.

Evaluation Following previous work (Luo et al., 2019; Sancheti et al., 2020), we assess style strength and content preservation. We fine-tune mBERT (Devlin et al., 2019) with Briakou et al. (2021b)'s pseudo-parallel corpora to evaluate the style accuracy of the outputs. We also use a style regressor from Briakou et al. (2021a), which is based on XLM-R (Conneau et al., 2020) and is shown to correlate well with human judgments.[7] We calculate BLEU and COMET (Rei et al., 2020) to assess content preservation. As overall score, following previous work, we compute the harmonic mean (HM) of style accuracy and BLEU.

Systems Based on our data (Section 2), we have four settings for our systems. **D1**: pseudo-parallel data in the target language via machine translating the English resource; **D2**: non-parallel style data in the target language; **D3**: no style data in the target language; **D4**: no parallel data at all. The first three settings all contain gold English parallel data.

Results Table 1 shows the results for both I→F (informal-to-formal) and F→I (formal-to-informal) transformations.[8] We include the models from Briakou et al. (2021b) for comparison (they only model the I→F direction).

[7] Results of classifiers/regressor are in Appendix A.2.

[8] Complete results are in Appendix A.3.

DATA	MODEL	INFORMAL→FORMAL									FORMAL→INFORMAL								
		ITALIAN			FRENCH			PORTUGUESE			ITALIAN			FRENCH			PORTUGUESE		
		BLEU	ACC	HM	BLEU	ACC	HM	BLEU	ACC	HM	BLEU	ACC	HM	BLEU	ACC	HM	BLEU	ACC	HM
D1	Multi-Task (Briakou et al., 2021b)	0.426	0.727	0.537	0.480	0.742	0.583	0.550	0.782	0.645	-	-	-	-	-	-	-	-	-
	M1.1: pseudo-parallel data	0.459	**0.856**	**0.598**	**0.530**	0.829	0.647	0.524	**0.852**	0.649	0.177	0.311	0.226	**0.195**	0.377	0.257	**0.225**	0.306	**0.259**
	M1.2: M1.1 + EN data	**0.461**	0.841	0.596	0.525	**0.863**	**0.653**	**0.553**	0.809	**0.657**	**0.178**	**0.315**	**0.227**	0.194	**0.458**	**0.273**	0.219	**0.313**	0.258
D2	DLSM (Briakou et al., 2021b)	0.124	0.223	0.159	0.180	0.152	0.165	0.185	0.191	0.188	-	-	-	-	-	-	-	-	-
	M2.1: IBT training + EN data	0.460	0.510	0.484	0.500	0.487	0.492	0.491	0.428	0.457	0.168	0.420	0.240	0.196	0.235	0.214	**0.237**	0.083	0.123
	M2.2: ADAPT + EN cross-attn	0.467	0.637	0.539	0.516	0.627	0.566	0.499	0.365	0.422	0.175	0.672	0.278	**0.212**	**0.627**	**0.317**	**0.237**	0.471	**0.315**
	M2.3: ADAPT + EN data	**0.476**	**0.731**	**0.577**	**0.519**	**0.702**	**0.597**	**0.526**	**0.509**	**0.517**	**0.180**	**0.719**	**0.288**	0.209	0.567	0.305	0.169	**0.534**	0.257
D3	M3.1: EN data	**0.485**	0.670	**0.563**	**0.553**	0.727	0.628	0.039	**0.890**	0.074	**0.186**	**0.767**	**0.299**	**0.216**	**0.692**	**0.329**	0.020	0.403	0.038
	M3.2: ADAPT + EN cross-attn	0.480	0.672	0.560	0.545	**0.749**	**0.631**	**0.547**	0.559	**0.553**	0.179	0.421	0.251	0.209	0.685	0.320	0.175	**0.560**	0.267
	M3.3: ADAPT + EN data	0.423	**0.735**	0.537	0.547	0.722	0.622	0.423	0.508	0.462	0.169	0.733	0.275	0.205	0.584	0.303	**0.189**	0.505	**0.275**
D4	Rule-based (Briakou et al., 2021b)	**0.438**	**0.268**	**0.333**	**0.472**	**0.208**	**0.289**	**0.535**	**0.448**	**0.488**	-	-	-	-	-	-	-	-	-
	M4.1: original mBART	0.380	0.103	0.162	0.425	0.080	0.135	0.128	0.200	0.156	0.160	**0.146**	**0.153**	0.189	**0.189**	**0.189**	0.080	**0.657**	**0.143**
	M4.2: ADAPT (generic data)	0.401	0.092	0.150	0.444	0.075	0.128	0.463	0.223	0.301	**0.164**	0.130	0.145	**0.194**	0.170	0.181	**0.237**	0.082	0.122

Table 1: Results for multilingual formality transfer. Notes: (i) for F→I there are four different source sentences and a human reference only, so for each instance scores are averaged; (ii) bold numbers denote best systems for each block, and underlined denote the best score for each transfer direction for each language.

Results in **D1** show that fine-tuning mBART with pseudo-parallel data yields the best overall performance in the I→F direction. The F→I results, instead, are rather poor and on Italian even worse than IBT-based models (M2.1). This could be due to this direction being harder in general, since there is more variation in informal texts, but it could also be made worse by the bad quality of the informal counterpart in the translated pairs. Indeed, work in machine translation has shown that low-quality data is more problematic in the target side than in the source side (Bogoychev and Sennrich, 2019).

In **D2**, we see that our proposed adaptation approaches outperform IBT-based models on both transfer directions. The results of fine-tuning the target language's model with English parallel data are generally better than inserting the EN model's cross-attention module into the target language's model. This suggests that the former can better transfer task and domain knowledge.

In **D3**, the large amounts of generic texts yield more improvement in I→F direction rather than F→I. This could be due to generic texts being more formal than informal. The performance improvement on Portuguese is particularly noticeable (compare M3.1 trained with EN data only with other M3.X models), and mostly due to this language being less represented than the others in mBART. Interestingly, the performance of task adaptation strategies is reversed compared to D2: it is here better to adapt cross attention in the English model rather than fine-tune the target language model directly. Future work will need to investigate how using different data sources for language adaptation (D2, style-specific vs D3, generic) interacts with task adaptation strategies.

Results for **D4** show that language adaptation training helps with content preservation, especially for Portuguese, confirming this curbs the problem of language underrepresentation in pre-training. However, low performance on style accuracy shows that task-specific data is necessary, even if it comes from a different language.

5 Analysis and Discussion

Case Study Table 2 shows a group of example outputs in Italian. In the I→F direction, most systems tend to copy a lot from the source and change formality words slightly. DLSM and Rule-based systems fail to transfer the formality style while others are successful to some extent: our M1.1 yields the best performance on the style strength. When looking at content, most outputs contain more or less part of the source sentence; Multi-Task system achieves the highest BLEU score but our systems (except for M3.3) have higher COMET scores, with M3.1 achieving the highest score. For the F→I direction, we can see that M1.1 has the worst performance on style strength (its output is almost identical to the source), while M2.1, M3.1 and M3.2 generate the same output with the lowest regression score. Overall, M3.3 achieves the best performance on style and content.

Direction Analysis For English, Rao and Tetreault (2018) find that the I→F direction is quite different from the opposite one since there are far more ways to express informality. As our work is the first attempt at the F→I direction in a multilingual setting, we run some additional analysis using two test sets for each direction: (a) the original test set; (b) the test set of the opposite direction, swapping sources and references. We fine-tune BART (Lewis et al., 2020) and mBART-50 (Tang et al., 2020) with English parallel data (GYAFC)

MODEL	SENTENCE	REG.	ACC	BLEU	COMET
	INFORMAL→FORMAL				
Source	se te ne vai secondo me e segno di debolezza e di paura se hai tanti problemi qui cerca di risolverli *if you go away I think it's a sign of weakness and fear if you have many problems here try to solve them*	-	-	-	-
Reference	Secondo il mio parere, il tuo andartene denota debolezza e paura, poiché se hai molti problemi, è necessario risolverli. *In my opinion, your going away denotes weakness and fear, since if you have many problems it is crucial to solve them.*	-	-	-	-
Multi-Task	Se te ne vai secondo me e segno di debolezza e di paura, se hai molti problemi qui, cerca di risolverli.	0.120	0.959	**0.151**	0.175
DLSM	Se te ne vai qualcosa e stesso di cui e di peggio se hai messo due soldi <unk> tutti i <unk> di <unk>	-2.666	0.014	0.015	-1.563
Rule-based	Se te ne vai secondo me e segno di debolezza e di paura se hai tanti problemi qui cerca di risolverli	-1.340	0.430	0.029	0.423
M1.1	Secondo me, è segno di debolezza e di paura. Se hai tanti problemi qui, cerca di risolverli.	**0.742**	**0.995**	0.035	0.658
M2.1	Se te ne vai secondo me e segno di debolezza e di paura. Se hai tanti problemi qui cerca di risolverli.	-0.243	0.978	0.028	0.634
M3.1	Se te ne vai, secondo me è segno di debolezza e di paura. Se hai tanti problemi, cerca di risolvere i problemi.	0.310	0.992	0.026	**0.728**
M3.2	Se te ne vai è segno di debolezza e di paura, se hai tanti problemi qui cerca di risolverli.	-0.225	0.971	0.037	0.639
M3.3	Its segno di debolezza e paura, se hai tanti problemi qui cerca di risolvere.	-0.092	0.692	0.126	-0.968
	FORMAL→INFORMAL				
Source	Se scrivi in italiano corretto avrai più possibilità di ricevere una risposta. *If you write in correct Italian you will have a better chance of receiving an answer.*	-	-	-	-
Reference	se magari scrivi in italiano riusciamo a risponderti!!! *maybe if you write in Italian we can answer you !!!*	-	-	-	-
M1.1	Se scrivi in italiano correttamente, avrai più possibilità di ottenere una risposta.	1.580	0.001	0.071	**0.566**
M2.1	se scrivi in italiano corretto avrai più possibilità di ricevere una risposta.	**0.221**	0.896	0.083	0.557
M3.1	se scrivi in italiano corretto avrai più possibilità di ricevere una risposta.	**0.221**	0.796	0.083	0.557
M3.2	se scrivi in italiano corretto avrai più possibilità di ricevere una risposta.	**0.221**	0.796	0.083	0.557
M3.3	scrivi in italiano e avrai più possibilità di ricevere una risposta.	0.891	**0.878**	**0.084**	**0.566**

Table 2: Example outputs in Italian and their sentence-level evaluation scores. Notes: (i) REG. indicates the score of the style regressor; (ii) ACC is the style confidence from the style classifier.

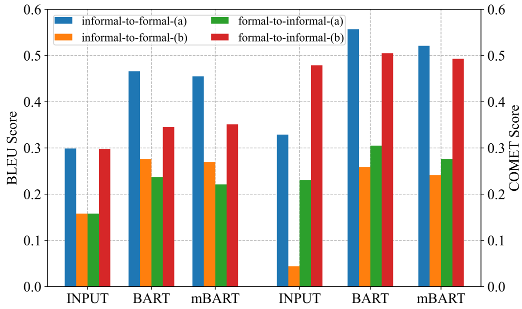


Figure 2: English formality transfer on content preservation using one reference. Setting (a) uses the original test set for each direction; (b) uses the test set of the opposite direction, swapping sources and references.

and evaluate them on (a) and (b). Figure 2 shows the results of content preservation. For INPUT (source copy), BLEU scores are almost the same swapping sources and references but COMET ones are not, probably due to COMET being trained to prefer a formal/better "generated sentence"; compared to INPUT, the performance gain of BART and mBART in I→F is larger than the opposite direction on both metrics. Results are similar for other languages (Table 3). We pick M1.1 and M1.2 from Table 1 since they are both fine-tuned using parallel data in the target language. BLEU scores of F→I are always lower than the opposite; the COMET score of INPUT in F→I is higher than I→F, but scores of both systems for F→I drop after transforming the source sentence into the target style. All these observations suggest that there is more variation in informal texts for the languages we consider, and the F→I direction is harder.

MODEL	ITALIAN		FRENCH		PORTUGUESE	
	BLEU	COMET	BLEU	COMET	BLEU	COMET
	INFORMAL→FORMAL (setting (a))					
INPUT	0.176	0.078	0.198	-0.019	0.244	0.217
M1.1	0.196	0.170	0.234	0.133	0.269	0.282
M1.2	0.194	0.181	0.231	0.138	0.283	0.319
	FORMAL→INFORMAL (setting (b))					
INPUT	0.174	0.364	0.196	0.277	0.243	0.463
M1.1	0.194	0.326	0.201	0.239	0.226	0.371
M1.2	0.193	0.311	0.199	0.219	0.220	0.358

Table 3: Results for multilingual formality transfer on content preservation using one reference.

6 Conclusions

Fine-tuning a pre-trained multilingual model with machine translated training data yields state-of-the-art results for transferring informal to formal text. The results for the formal-to-informal direction are considerably worse—the task is more difficult, and the quality of translated informal text is lower. We have also proposed two adaptation training strategies that can be applied in a cross-lingual transfer strategy . These strategies target language and task adaptation, and can be combined to adapt mBART for multilingual formality transfer. The adaptation strategies with auxiliary parallel data from a different language are effective, yielding competitive results and outperforming more classic IBT-based approaches without task-specific parallel data. Lastly, we have shown that formal-to-informal transformation is harder than the opposite direction.

Acknowledgments

This work was partly funded by the China Scholarship Council (CSC). The anonymous reviewers of ACL Rolling Review provided us with useful comments which contributed to improving this paper and its presentation, so we're grateful to them. We would also like to thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Peregrine high performance computing cluster.

Ethics Statement

All work that automatically generates and/or alters natural text could unfortunately be used maliciously. While we cannot fully prevent such uses once our models are made public, we do hope that writing about risks explicitly and also raising awareness of this possibility in the general public are ways to contain the effects of potential harmful uses. We are open to any discussion and suggestions to minimise such risks.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Nikolay Bogoychev and Rico Sennrich. 2019. Domain, translationese and noise in synthetic data for neural machine translation. *arXiv preprint arXiv:1911.03362*.

Eleftheria Briakou, Sweta Agrawal, Joel Tetreault, and Marine Carpuat. 2021a. Evaluating the evaluation metrics for style transfer: A case study in multilingual formality transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1321–1336, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Eleftheria Briakou, Di Lu, Ke Zhang, and Joel Tetreault. 2021b. Olá, bonjour, salve! XFORMAL: A benchmark for multilingual formality style transfer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3199–3216, Online. Association for Computational Linguistics.

Yixin Cao, Ruihao Shui, Liangming Pan, Min-Yen Kan, Zhiyuan Liu, and Tat-Seng Chua. 2020. Expertise style transfer: A new task towards better communication between experts and laymen. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1061–1071, Online. Association for Computational Linguistics.

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

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

Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. 2021. XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703, Online. Association for Computational Linguistics.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, Melbourne, Australia. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*.

Huiyuan Lai, Antonio Toral, and Malvina Nissim. 2021a. Generic resources are what you need: Style transfer tasks without task-specific parallel training data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4241–4254, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Huiyuan Lai, Antonio Toral, and Malvina Nissim. 2021b. Thank you BART! rewarding pre-trained models improves formality style transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 484–494, Online. Association for Computational Linguistics.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019. Multiple-attribute text rewriting. In *International Conference on Learning Representations*.

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

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana. Association for Computational Linguistics.

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

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Zhifang Sui, and Xu Sun. 2019. A dual reinforcement learning framework for unsupervised text style transfer. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5116–5122.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876, Melbourne, Australia. Association for Computational Linguistics.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 129–140, New Orleans, Louisiana. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Abhilasha Sancheti, Kundan Krishna, Balaji Vasan Srinivasan, and Anandhavelu Natarajan. 2020. Reinforced rewards framework for text style transfer. In *Advances in Information Retrieval*, pages 545–560.

Asa Cooper Stickland, Alexandre Bérard, and Vassilina Nikoulina. 2021a. Multilingual domain adaptation for nmt: Decoupling language and domain information with adapters. *arXiv preprint, arXiv: 2110.09574*.

Asa Cooper Stickland, Xian Li, and Marjan Ghazvininejad. 2021b. Recipes for adapting pre-trained monolingual and multilingual models to machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3440–3453, Online. Association for Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint, arXiv: 2008.00401*.

Ahmet Üstün, Alexandre Berard, Laurent Besacier, and Matthias Gallé. 2021. Multilingual unsupervised neural machine translation with denoising adapters. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6650–6662, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly Universal Dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2302–2315, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xiaoyuan Yi, Zhenghao Liu, Wenhao Li, and Maosong Sun. 2020. Text style transfer via learning style instance supported latent space. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3801–3807.

Yi Zhang, Tao Ge, and Xu Sun. 2020. Parallel data augmentation for formality style transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3221–3228, Online. Association for Computational Linguistics.

A Appendices:

This appendices include: (i) Results for BART and mBART on English data (A.1); (ii) Results for style classifiers/regressor (A.2); (iii) Detailed results for multilingual formality transfer (A.3).

A.1 Results for BART and mBART on English data

We fine-tune BART (Lewis et al., 2020) and mBART-50 (Tang et al., 2020) with English parallel data specifically developed for formality transfer in English (GYAFC). The performance of BART and English data can be seen as a sort of upperbound, as these are best conditions (monolingual model, and gold parallel data). The drop we see using mBART is rather small, suggesting mBART is a viable option. We also see that formal to informal is much harder than viceversa, probably due to high variability in informal formulations (Rao and Tetreault, 2018). Table A.1 shows the results for both models.

MODEL	DIRECTION	COMET	BLEU	REG.	ACC	HM
BART	Informal→Formal	0.544	0.795	-0.527	0.928	0.856
	Formal→Informal	0.170	0.436	-1.143	0.683	0.532
mBART	Informal→Formal	0.512	0.779	-0.531	0.916	0.842
	Formal→Informal	0.151	0.422	-1.031	0.591	0.492

Table A.1: Results of BART and mBART on English data. Note that REG. indicates the score of the style regressor (the higher is better in Informal→Formal, lower is better in Formal→Informal).

A.2 Results for style classifiers/regressor

We compare four different style classifiers and one regressor: (i) TextCNN (Kim, 2014) trained with pseudo-parallel data in the target language; (ii) mBERT (Devlin et al., 2019) fine-tuned with pseudo-parallel data, English data, or a combination of all data; and (iii) a XLM-R (Conneau et al., 2020) based style regressor from Briakou et al. (2021a), which is trained with formality rating data in English.

MODEL	TRAINING DATA	ITALIAN				FRENCH				PORTUGUESE			
		ACC	Precision	Recall	F1	ACC	Precision	Recall	F1	ACC	Precision	Recall	F1
TextCNN	Pseudo data	0.865	0.885	0.839	0.861	0.838	0.876	0.787	0.829	0.799	0.793	0.809	0.801
mBERT	Pseudo data	**0.898**	0.905	0.890	**0.897**	0.879	**0.918**	0.831	0.872	**0.851**	0.806	0.924	**0.861**
mBERT	English data	0.889	0.856	**0.934**	0.893	**0.896**	0.856	**0.951**	**0.901**	0.839	0.771	**0.964**	0.857
mBERT	All data	0.891	**0.906**	0.872	0.888	0.882	0.911	0.846	0.877	**0.851**	**0.815**	0.909	0.859
XLM-R	Formality ratings	Informal: -1.672		Formal: 0.108		Informal: -1.701		Formal: 0.050		Informal: -1.438		Formal: 0.065	

Table A.2: Results for style classifiers/regressor on test set. The data used for evaluation are 1000 sentences from the test set and the corresponding 1000 human references. For informal sentences, the smaller the XLM-R score is better, higher is better for formal sentences.

A.3 Detailed results for multilingual formality transfer

Data	Model	Italian					French					Portuguese				
		COMET	BLEU	REG.	ACC	HM	COMET	BLEU	REG.	ACC	HM	COMET	BLEU	REG.	ACC	HM
Transfer Direction: Informal→Formal																
D1	Translate Train Tag (Briakou et al., 2021b)	-0.059	0.426	-0.705	0.735	0.539	-0.164	0.451	-0.586	0.696	0.547	0.194	0.524	-0.636	0.755	0.619
	+ Back-Tranlated Data (Briakou et al., 2021b)	0.026	0.430	-0.933	0.556	0.485	0.004	0.491	-0.898	0.485	0.488	0.301	0.546	-0.875	0.627	0.584
	Multi-Task Tag-Style (Briakou et al., 2021b)	-0.021	0.426	-0.698	0.727	0.537	-0.062	0.480	-0.501	0.742	0.583	0.266	0.550	-0.578	0.782	0.645
	M1.1: pseudo-parallel data	0.143	0.459	-0.426	**0.856**	**0.598**	0.124	**0.530**	-0.305	0.829	0.647	0.297	0.524	**-0.334**	**0.852**	0.649
	M1.2: M1.1 + EN parallel data	**0.147**	**0.461**	-0.442	0.841	0.596	**0.130**	0.525	-0.275	**0.863**	**0.653**	**0.331**	**0.553**	-0.395	0.809	**0.657**
	M1.3: all data (one model)	0.137	**0.461**	**-0.409**	0.850	**0.598**	0.127	0.515	**-0.267**	0.851	0.642	0.309	0.537	-0.367	0.803	0.644
D2	DLSM (Briakou et al., 2021b)	-1.332	0.124	-2.141	0.223	0.159	-1.267	0.180	-2.021	0.152	0.165	-1.131	0.185	-2.078	0.191	0.188
	M2.1: IBT training	0.057	0.420	-1.351	0.240	0.305	-0.019	0.465	-1.303	0.219	0.298	0.233	0.487	-1.074	0.411	0.446
	M2.2: M2.1 + EN data	0.105	0.460	-0.867	0.510	0.484	0.036	0.500	-0.814	0.487	0.492	0.236	0.491	-1.040	0.428	0.457
	M2.3: ADAPT + EN cross-attn	**0.139**	0.467	-0.684	0.637	0.539	0.066	0.516	-0.613	0.627	0.566	0.288	0.499	-1.143	0.365	0.422
	M2.4: ADAPT + EN data	0.131	**0.476**	**-0.537**	**0.731**	**0.577**	**0.074**	**0.519**	**-0.572**	**0.702**	**0.597**	**0.291**	**0.526**	**-0.922**	**0.509**	**0.517**
D3	M3.1: EN data	**0.134**	**0.485**	-0.590	0.670	**0.563**	**0.102**	**0.553**	-0.591	0.727	0.628	-1.673	0.039	**-0.550**	**0.890**	0.074
	M3.2: ADAPT + EN cross-attn	0.130	0.480	-0.588	0.672	0.560	0.091	0.545	**-0.446**	**0.749**	**0.631**	**0.302**	**0.547**	-0.859	0.559	**0.553**
	M3.3: ADAPT + EN data	-0.107	0.423	**-0.579**	**0.735**	0.537	0.101	0.547	-0.488	0.722	0.622	-0.260	0.423	-1.112	0.508	0.462
D4	Round-trip MT (Briakou et al., 2021b)	-0.053	0.346	**-1.026**	**0.354**	**0.350**	-0.065	0.416	**-0.748**	**0.406**	**0.411**	0.213	0.430	**-0.661**	**0.601**	**0.501**
	Rule-based (Briakou et al., 2021b)	**0.071**	**0.438**	-1.167	0.268	0.333	**-0.013**	**0.472**	-1.236	0.208	0.289	**0.291**	**0.535**	-1.081	0.448	0.488
	M4.1: original mBART	-0.067	0.380	-1.672	0.103	0.162	-0.106	0.425	-1.709	0.080	0.135	-1.444	0.128	-1.870	0.200	0.156
	M4.3: ADAPT (generic data)	0.033	0.401	-1.675	0.092	0.150	-0.033	0.444	-1.700	0.075	0.128	0.230	0.463	-1.438	0.223	0.301
Transfer Direction: Formal→Informal																
D1	M1.1: pseudo-parallel data	**0.298**	0.177	-0.225	0.311	0.226	**0.239**	**0.195**	-0.188	0.377	0.257	0.388	0.225	-0.273	0.306	**0.259**
	M1.2: M1.1 + EN parallel data	0.278	**0.178**	-0.228	0.315	0.227	0.215	0.194	**-0.304**	**0.458**	**0.273**	0.373	0.219	**-0.282**	**0.313**	0.258
	M1.3: all data (one model)	0.283	0.175	**-0.287**	**0.368**	**0.237**	0.207	0.191	-0.301	0.439	0.266	**0.407**	**0.229**	-0.241	0.292	0.257
D2	M2.1: IBT training	0.335	0.166	-0.082	0.338	0.223	0.272	0.195	0.037	0.194	0.194	0.467	**0.237**	0.042	0.084	0.124
	M2.2: M2.1 + EN data	**0.337**	0.168	-0.174	0.420	0.240	**0.274**	0.196	-0.016	0.235	0.214	**0.471**	**0.237**	0.045	0.083	0.123
	M2.3: ADAPT + EN cross-attn	0.176	0.175	**-0.631**	0.672	0.278	0.226	**0.212**	**-0.464**	**0.627**	**0.317**	0.441	**0.237**	-0.343	0.471	**0.315**
	M2.4: ADAPT + EN data	0.279	**0.180**	-0.582	**0.719**	**0.288**	0.232	0.209	-0.444	0.567	0.305	-0.022	0.169	**-0.520**	**0.534**	0.257
D3	M3.1: EN data	0.289	**0.186**	-0.646	**0.767**	**0.299**	**0.244**	**0.216**	-0.566	**0.692**	**0.329**	-1.695	0.020	**-1.225**	0.403	0.038
	M3.2: ADAPT + EN cross-attn	**0.300**	0.179	-0.285	0.421	0.251	0.221	0.209	**-0.594**	0.685	0.320	**0.367**	0.175	-0.449	**0.560**	0.267
	M3.3: ADAPT + EN data	0.100	0.169	**-0.744**	0.733	0.275	0.220	0.205	-0.447	0.584	0.303	0.130	**0.189**	-0.586	0.505	**0.275**
D4	M4.1: original mBART	0.260	0.160	**0.076**	**0.146**	**0.153**	0.204	0.189	**0.031**	**0.189**	**0.189**	-1.363	0.080	**-1.406**	**0.657**	**0.143**
	M4.2: ADAPT (generic data)	**0.317**	**0.164**	0.084	0.130	0.145	**0.268**	**0.194**	0.052	0.170	0.181	**0.475**	**0.237**	0.047	0.082	0.122

Table A.3: Results for multilingual formality transfer. Notes: (i) REG. indicates the score of the style regressor (the higher is better in I→F, lower is better in F→I); (ii) for F→I there are four different source sentences and a human reference only, so for each instance scores are averaged; (iii) bold numbers denote best systems for each block, and underlined indicate the best score for each transfer direction.

When to Use Multi-Task Learning vs Intermediate Fine-Tuning for Pre-Trained Encoder Transfer Learning

Orion Weller*
Johns Hopkins University

Kevin Seppi
Brigham Young University

Matt Gardner
Microsoft Semantic Machines

Abstract

Transfer learning (TL) in natural language processing (NLP) has seen a surge of interest in recent years, as pre-trained models have shown an impressive ability to transfer to novel tasks. Three main strategies have emerged for making use of multiple supervised datasets during fine-tuning: training on an intermediate task before training on the target task (STILTs), using multi-task learning (MTL) to train jointly on a supplementary task and the target task (pairwise MTL), or simply using MTL to train jointly on all available datasets (MTL$_{\text{All}}$). In this work, we compare all three TL methods in a comprehensive analysis on the GLUE dataset suite. We find that there is a simple heuristic for when to use one of these techniques over the other: pairwise MTL is better than STILTs when the target task has fewer instances than the supporting task and vice versa. We show that this holds true in more than 92% of applicable cases on the GLUE dataset and validate this hypothesis with experiments varying dataset size. The simplicity and effectiveness of this heuristic is surprising and warrants additional exploration by the TL community. Furthermore, we find that MTL$_{\text{All}}$ is worse than the pairwise methods in almost every case. We hope this study will aid others as they choose between TL methods for NLP tasks. [1]

1 Introduction

The standard supervised training paradigm in NLP research is to fine-tune a pre-trained language model on some target task (Peters et al., 2018; Devlin et al., 2018; Raffel et al., 2019; Gururangan et al., 2020). When additional non-target supervised datasets are available during fine-tuning, it is not always clear how to best make use of the supporting data (Phang et al., 2018, 2020; Liu et al., 2019b,a; Pruksachatkun et al., 2020a). Although there are an exponential number of ways to combine or alternate between the target and supporting tasks, three predominant methods have emerged: (1) fine-tuning on a supporting task and then the target task consecutively, often called STILTs (Phang et al., 2018); (2) fine-tuning on a supporting task and the target task simultaneously (here called pairwise multi-task learning, or simply MTL); and (3) fine-tuning on all N available supporting tasks and the target tasks together (MTL$_{\text{All}}$, $N > 1$).

Application papers that use these methods generally focus on only one method (Søgaard and Bingel, 2017; Keskar et al., 2019; Glavas and Vulić, 2020; Sileo et al., 2019; Zhu et al., 2019; Weller et al., 2020; Xu et al., 2019; Chang and Lu, 2021), while a limited amount of papers consider running two. Those that do examine them do so with a limited number of configurations: Phang et al. (2018) examines STILTS and one instance of MTL, Changpinyo et al. (2018); Peng et al. (2020); Schröder and Biemann (2020) compare MTL with MTL$_{\text{All}}$, and Wang et al. (2018a); Talmor and Berant (2019); Liu et al. (2019b); Phang et al. (2020) use MTL$_{\text{All}}$ and STILTs but not pairwise MTL.

In this work we perform comprehensive experiments using all three methods on the 9 datasets in the GLUE benchmark (Wang et al., 2018b). We surprisingly find that a simple size heuristic can be used to determine with more than 92% accuracy which method to use for a given target and supporting task: when the target dataset is larger than the supporting dataset, STILTS should be used; otherwise, MTL should be used (MTL$_{\text{All}}$ is almost universally the worst of the methods in our experiments). To confirm the validity of the size heuristic, we additionally perform a targeted experiment varying dataset size for two of the datasets, showing that there is a crossover point in performance between the two methods when the dataset sizes are equal. We believe that this analysis will help NLP researchers to make better decisions when choosing

[1] We make our code publicly available at https://github.com/orionw/MTLvsIFT.

* Corresponding author, oweller2@jhu.edu

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 272 - 282
May 22-27, 2022 ©2022 Association for Computational Linguistics

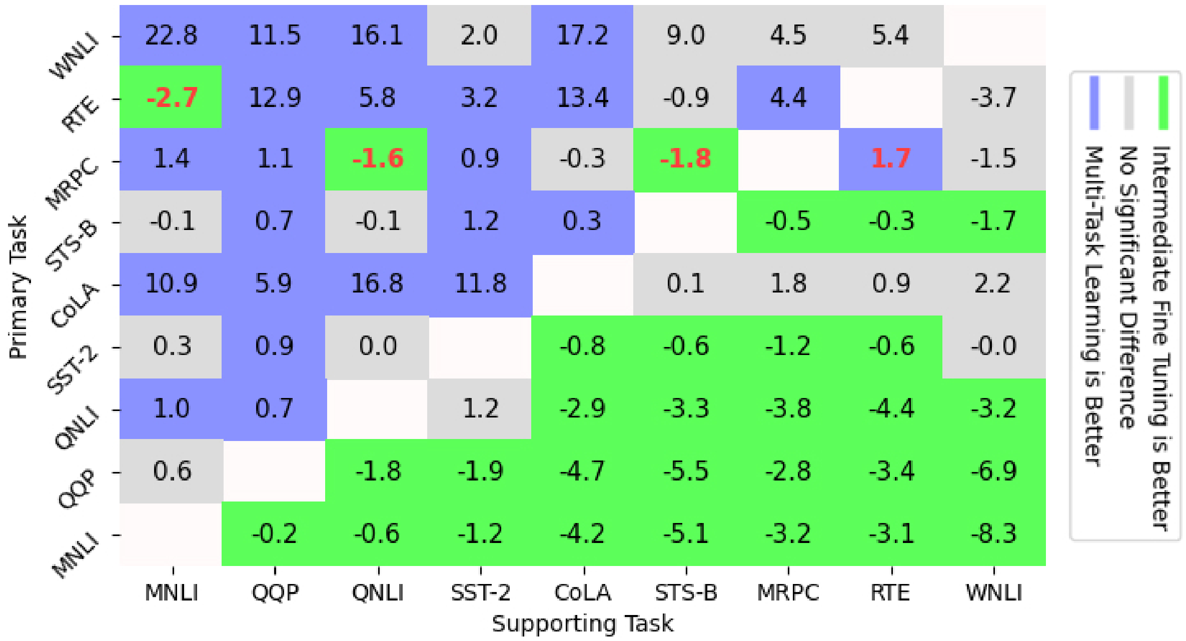


Figure 1: Results comparing intermediate fine tuning (STILTs) vs multi-task learning (MTL). Numbers in cells indicate the absolute percent score difference on the primary task when using MTL instead of STILTs (positive scores mean MTL is better and vice versa). The colors indicate visually the best method, showing a statistically significant difference from the other from using using a two-sided t-test with $\alpha = 0.1$. Numbers in red indicate the cells where the size heuristic does not work. Datasets are ordered in descending size (WNLI is the smallest).

a TL method and will open up future research into understanding the cause of this heuristic's success.

2 Experimental Settings

Dataset Suite To conduct this analysis, we chose to employ the GLUE dataset suite, following and comparing to previous work in transfer learning for NLP (Phang et al., 2018; Liu et al., 2019b).

Training Framework We use Huggingface's *transformers* library (Wolf et al., 2019) for accessing the pre-trained encoder and for the base training framework. We extend this framework to combine multiple tasks into a single PyTorch (Paszke et al., 2017) dataloader for MTL and STILTs training.

Many previous techniques have been proposed for how to best perform MTL (Raffel et al., 2019; Liu et al., 2019b), but a recent paper by Gottumukkala et al. (2020) compared the main approaches and showed that a new dynamic approach provides the best performance in general. We implement all methods described in their paper and experimented with several approaches (sampling by size, uniformity, etc.). Our initial results found that dynamic sampling was indeed the most effective on pairwise tasks. Thus, for the remainder of this paper, our MTL framework uses dynamic sampling with heterogeneous batch schedules. For consistency, we train the STILTs models using the same code, but include only one task in the dataloader instead of multiple. The MTL_{All} setup uses the same MTL code, but includes all 9 GLUE tasks.

We train each model on 5 different seeds to control for randomness (Dodge et al., 2020). For the STILTs method, we train 5 models with different seeds on the supporting task and then choose the best of those models to train with 5 more random seeds on the target task. For our final reported numbers, we record both the average score and the standard deviation, comparing the MTL approach to the STILTs approach with a two-sample t-test. In total, we train $9 * 8 * 5 = 360$ different MTL versions of our model, 5 MTL_{All} models, and $9 * 5 + 9 * 5 = 90$ models in the STILTs setting.

Model We use the DistilRoBERTa model (pretrained and distributed from the *transformers* library similarly to the DistilBERT model in Sanh et al. (2019)) for our experiments, due to its strong performance and efficiency compared to the full model. For details regarding model and compute parameters, see Appendix A. Our purpose is *not* to train the next state-of-the-art model on the GLUE task and thus the absolute scores are not immediately relevant; our purpose is to show how the different methods score *relative to each other*. We note that we conducted the same analysis in Fig-

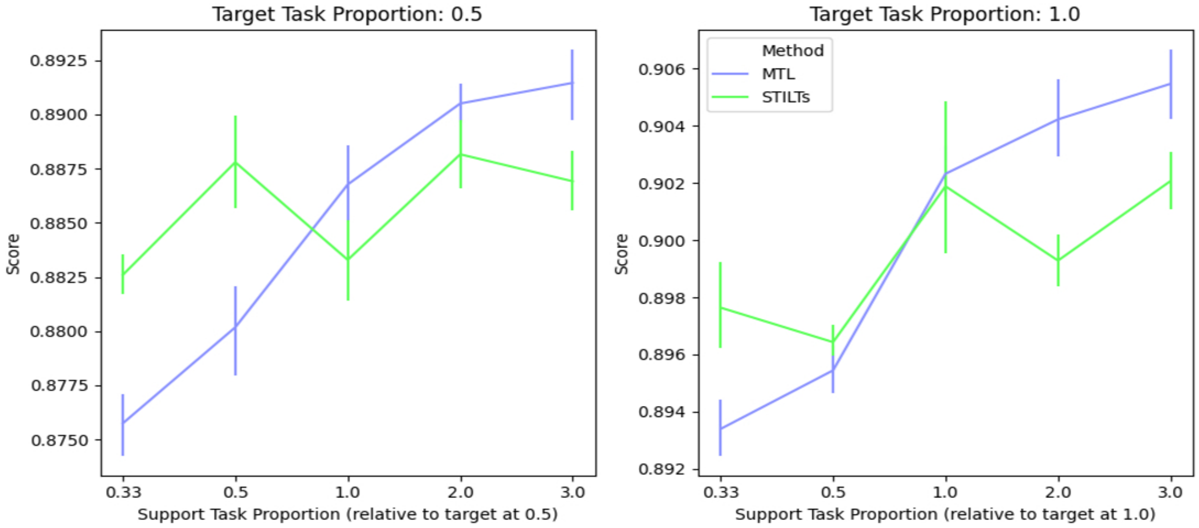


Figure 2: Experiments validating the size heuristic on the (QNLI, MNLI) task pair. The right figure shows training on 100% of the QNLI training set while the left figure shows training with 50%. The x-axis indicates the amount of training data of the supporting task (MNLI) relative to the QNLI training set, artificially constrained (e.g. 0.33 indicates that the supporting task is a third of the size of the QNLI training set, etc.). The blue line indicates MTL results while the green line indicates the STILTs method. Error bars indicate a 90% CI using 5 random seeds.

ure 1 for BERT and found the same conclusion (see Appendix D), showing that our results extend to other pre-trained transformers.

3 Results

We provide three different analyses: a comparison of pairwise MTL vs STILTs, experiments varying dataset size to validate our findings, and a comparison of pairwise approaches vs MTL_{All}.

MTL vs STILTs We first calculate the absolute score matrices from computing the MTL and STILTs method on each pair of the GLUE dataset suite, then subtract the STILTs average score matrix from the MTL one (Figure 1). Thus, this shows the absolute score gain for using the MTL method instead of the STILTs method (negative scores indicate that the STILTs method was better, etc.).

However, this matrix does not tell us whether these differences are statistically significant; for this we use a two-sample t-test to compare the mean and standard deviation of each method for a particular cell. Scores that are statistically significant are color coded green (if STILTs is better) or blue (if MTL is better), whereas they are coded grey if there is no statistically significant difference. We note that although some differences are large (e.g. a 9 point difference on (WNLI, STS-B)) the variance of these results is high enough that there is no statistically significant difference between the STILTs and MTL score distributions.

We order the datasets in Figure 1 by size, to visually illustrate the trend. The number of green cells in a row is highly correlated with the size of the dataset represented by that row. For example, MNLI is the largest and every cell in the MNLI row is green. QQP is the 2nd largest and every cell in its row is also green, except for (QQP, MNLI). The smallest dataset, WNLI, has zero green cells.

We can summarize these results with the following size heuristic: **MTL is better than STILTs when the target task has fewer training instances than the supporting task** and vice versa. In fact, if we use this heuristic to predict which method will be better we find that it predicts 49/53 significant cells, which is equivalent to 92.5% accuracy. To more clearly visualize which cells it fails to predict accurately, those four cells are indicated with red text. We note that this approach does not hold on the cells that have no statistically significant difference between the two methods: but for almost every significant cell, it does.

Unfortunately, there is no clear answer to why those four cells are misclassified. Three of the four misclassified cells come when using the MRPC dataset as the target task, but there is no obvious reason why it fails on MRPC. We recognize that this size heuristic is not an absolute law, but merely a good heuristic that does so with high accuracy: there are still other pieces to this puzzle that this work does not consider, such as dataset similarity.

Dataset Size Experiments In order to validate

Approach	Mean	WNLI	STS-B	SST-2	RTE	QQP	QNLI	MRPC	MNLI	CoLA
MTL$_{\text{All}}$	73.3	54.4	86.6	90.8	**67.4**	80.2	84.9	85.4	74.2	35.8
Avg. STILTs	75.8	45.0	87.5	92.1	61.9	88.9	89.4	**87.4**	**84.0**	46.4
Avg. MTL	77.3	**56.1**	87.4	91.9	66.0	85.6	87.5	**87.4**	80.8	**52.7**
Avg. S.H.	**78.3**	**56.1**	**87.7**	**92.3**	66.5	**89.0**	**89.6**	87.3	**84.0**	52.1
Pairwise Oracle	80.7	57.7	88.8	92.9	76.0	89.5	90.6	90.2	84.3	56.5

Table 1: Comparison of MTL$_{\text{All}}$ to the pairwise STILTs or MTL approaches. "S.H" stands for size heuristic. Pairwise Oracle uses the best supplementary task for the given target task using the best pairwise method (STILTs or MTL). All scores are the average of 5 random seeds. We find that on almost every task, pairwise approaches are better than MTL$_{\text{All}}$. Bold scores indicate the best score in the column, excluding the oracle.

the size heuristic further we conduct controlled experiments that alter the amount of training data of the supporting task to be above and below the target task. We choose to test QNLI primary with MNLI supporting, as they should be closely related and thus have the potential to disprove this heuristic. We subsample data from the supporting task so that we have a proportion K of the size of the primary task (where $K \in \{1/3, 1/2, 1, 2, 3\}$). By doing so, we examine whether the size heuristic holds while explicitly controlling for the supporting task's size. Other than dataset size, all experimental parameters are the same as in the original comparison (§2).

We also test whether these results hold if the size of the primary dataset is changed (e.g., perhaps there is something special about the current size of the QNLI dataset). We take the same pair and reduce the training set of QNLI in half, varying MNLI around the new number of instances in the QNLI training set as above (e.g. 1/3rd, 1/2, etc.).

The results of these two experiments are in Figure 2. We can see that as the size of the supporting dataset increases, MTL becomes more effective than STILTs. Furthermore, we find that when both datasets are equal sizes the two methods are statistically similar, as we would expect from the size heuristic (Support Task Proportion=1.0).

Thus, the synthetic experiments corroborate our main finding; the size heuristic holds even on controlled instances where the size of the training sets are artificially manipulated.

Pairwise TL vs MTL$_{\text{All}}$ We also experiment with MTL$_{\text{All}}$ on GLUE (see Appendix B for implementation details). We find that the average pairwise approach consistently outperforms the MTL$_{\text{All}}$ method, except for the RTE task (Table 1) and using the best supporting task outperforms MTL$_{\text{All}}$ in every case (Pairwise Oracle). Thus, although MTL$_{\text{All}}$ is conceptually simple, it is not the best choice w.r.t. the target task score: on a random dataset simply using STILTs or MTL will likely perform better. Furthermore, using the size heuristic on the average supplementary task increases the score by 5 points over MTL$_{\text{All}}$ (78.3 vs 73.3).

4 Related Work

A large body of recent work (Søgaard and Bingel, 2017; Vu et al., 2020; Bettgenhäuser et al., 2020; Peng et al., 2020; Poth et al., 2021) exists that examines *when* these transfer learning methods are more effective than simply fine-tuning on the target task. Oftentimes, these explanations involve recognizing catastrophic forgetting (Phang et al., 2018; Pruksachatkun et al., 2020b; Wang et al., 2018a) although recent work has called for them to be reexamined (Chang and Lu, 2021). This paper is orthogonal to those, as we examine when you should choose MTL or STILTs, rather than when they are more effective than the standard fine-tuning case (in fact, these strategies could be combined to predict transfer and then use the best method). As our task is different, theoretical explanations for how these methods work *in relation to each other* will need to be explored in future work. Potential theories suggested by our results are discussed in Appendix C, and are left to guide those efforts.

5 Conclusion

We examined the three main strategies for transfer learning in natural language processing: training on an intermediate supporting task to aid the target task (STILTs), training on the target and supporting task simultaneously (MTL), or training on multiple supporting tasks alongside the target task (MTL$_{\text{All}}$). We provide the first comprehensive comparison between these three methods using the GLUE dataset suite and show that there is a simple rule for when to use one of these techniques over the other. This simple heuristic, which holds true in more than 92% of applicable cases, states that multi-task learning is better than intermediate fine tuning when the target task is smaller than the supporting task and vice versa. Additionally, we showed that these pairwise transfer learning techniques outperform the $\mathrm{MTL}_{\mathrm{All}}$ approach in almost every case.

References

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth PASCAL recognizing textual entailment challenge.

Gabriele Bettgenhäuser, Michael A Hedderich, and Dietrich Klakow. 2020. Learning functions to study the benefit of multitask learning. *arXiv preprint arXiv:2006.05561*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Ting-Yun Chang and Chi-Jen Lu. 2021. Rethinking why intermediate-task fine-tuning works. *arXiv preprint arXiv:2108.11696*.

Soravit Changpinyo, Hexiang Hu, and Fei Sha. 2018. Multi-task learning for sequence tagging: An empirical study. *arXiv preprint arXiv:1808.04151*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *ArXiv*, abs/2003.10555.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the International Workshop on Paraphrasing*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9. Association for Computational Linguistics.

Goran Glavas and I. Vulić. 2020. Is supervised syntactic parsing beneficial for language understanding? an empirical investigation. *ArXiv*, abs/2008.06788.

Ananth Gottumukkala, Dheeru Dua, Sameer Singh, and Matt Gardner. 2020. Dynamic sampling strategies for multi-task reading comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 920–924.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*.

N. Keskar, Bryan McCann, Caiming Xiong, and R. Socher. 2019. Unifying question answering and text classification via span extraction. *ArXiv*, abs/1904.09286.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The Winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, page 47.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. *arXiv preprint arXiv:1904.09482*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019b. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Unicorn on rainbow: A universal commonsense reasoning model on a new multitask benchmark. *arXiv preprint arXiv:2103.13009*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Yifan Peng, Qingyu Chen, and Zhiyong Lu. 2020. An empirical study of multi-task learning on bert for biomedical text mining. *arXiv preprint arXiv:2005.02799*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Jason Phang, Phu Mon Htut, Yada Pruksachatkun, Haokun Liu, Clara Vania, Katharina Kann, Iacer Calixto, and Samuel R Bowman. 2020. English intermediate-task training improves zero-shot cross-lingual transfer too. *arXiv preprint arXiv:2005.13013*.

Clifton Poth, Jonas Pfeiffer, Andreas Ruckl'e, and Iryna Gurevych. 2021. What to pre-train on? efficient intermediate task selection. *ArXiv*, abs/2104.08247.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, C. Vania, K. Kann, and Samuel R. Bowman. 2020a. Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? *ArXiv*, abs/2005.00628.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R Bowman. 2020b. Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? *arXiv preprint arXiv:2005.00628*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Fynn Schröder and Chris Biemann. 2020. Estimating the influence of auxiliary tasks for multi-task learning of sequence tagging tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2971–2985.

Damien Sileo, Tim Van-de Cruys, Camille Pradel, and Philippe Muller. 2019. Discourse-based evaluation of language understanding. *arXiv preprint arXiv:1907.08672*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, pages 1631–1642.

Anders Søgaard and Joachim Bingel. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *EACL*.

Alon Talmor and Jonathan Berant. 2019. Multiqa: An empirical investigation of generalization and transfer in reading comprehension. *arXiv preprint arXiv:1905.13453*.

Nikos Voskarides, Dan Li, A. Panteli, and Pengjie Ren. 2019. Ilps at trec 2019 conversational assistant track. In *TREC*.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across nlp tasks. *arXiv preprint arXiv:2005.00770*.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, et al. 2018a. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. *arXiv preprint arXiv:1812.10860*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018b. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2018. Neural network acceptability judgments. *arXiv preprint 1805.12471*.

Orion Weller, Nicholas Lourie, Matt Gardner, and Matthew E. Peters. 2020. Learning from task descriptions. *ArXiv*, abs/2011.08115.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL-HLT*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yichong Xu, X. Liu, C. Li, Hoifung Poon, and Jianfeng Gao. 2019. Doubletransfer at mediqa 2019: Multi-source transfer learning for natural language understanding in the medical domain. In *BioNLP@ACL*.

Zeyu Yan, Jianqiang Ma, Y. Zhang, and Jianping Shen. 2020. Sql generation via machine reading comprehension. In *COLING*.

Approach	Mean	WNLI	STS-B	SST-2	RTE	QQP	QNLI	MRPC	MNLI	CoLA
MTL_{All} Uniform	63.2	**56.1**	85.1	84.0	58.3	70.4	76.4	80.3	50.7	7.8
MTL_{All} Dynamic	67.2	52.1	86.2	88.4	63.8	75.5	81.2	82.3	64.0	10.9
MTL_{All} Size	**73.3**	54.4	**86.6**	**90.8**	**67.4**	**80.2**	**84.9**	**85.4**	**74.2**	**35.8**
Avg. STILTs	75.8	45.0	87.5	92.1	61.9	88.9	89.4	**87.4**	**84.0**	46.4
Avg. MTL	77.3	**56.1**	87.4	91.9	66.0	85.6	87.5	**87.4**	80.8	**52.7**
Avg. S.H.	**78.3**	**56.1**	**87.7**	**92.3**	66.5	**89.0**	**89.6**	87.3	**84.0**	52.1
Pairwise Oracle	**80.7**	**57.7**	**88.8**	**92.9**	**76.0**	**89.5**	**90.6**	**90.2**	**84.3**	**56.5**

Table 2: Comparison of MTL_{All} to the pairwise STILTs or MTL approaches. “S.H" stands for size heuristic. Pairwise Oracle uses the best supplementary task for the given target task using the best pairwise method (STILTs or MTL). All scores are the average of 5 random seeds. Note that MTL_{All} was run with three different sampling methods (top half). We find that on almost every task, pairwise approaches are better than MTL_{All}. Bold scores indicate the best score in the column for the given section.

Wei Zhu, Xiaofeng Zhou, K. Wang, X. Luo, Xiepeng Li, Y. Ni, and G. Xie. 2019. Panlp at mediqa 2019: Pre-trained language models, transfer learning and knowledge distillation. In *BioNLP@ACL*.

A Training and Compute Details

We use the hyperparameters given by the *transformer* library example on GLUE as the default for our model (learning rate of 2e-5, batch size of 128, AdamW optimizer (Kingma and Ba, 2014), etc.). We train for 10 epochs, checkpointing every half an epoch and use the best model on the development set for the test set scores. We train on a mix of approximately 10 K80 and P100 GPUs for approximately two weeks for the main experiment, using another week of compute time for the synthetic experiments (§3). Our CPUs use 12-core Intel Haswell (2.3 GHz) processors with 32GB of RAM.

B Pairwise Approaches vs MTL_{All}

Experimental Setup We use MTL_{All} with three different sampling methods: uniform sampling, sampling by dataset size, and dynamic sampling. To illustrate the difference between MTL_{All} and the pairwise methods, we show the average score across all supplementary tasks for MTL and STILTs. We also show the average score found by choosing MTL or STILTs using the size heuristic as *Ave. S.H.*. Finally, we report the score from the best task using the best pairwise method, which we call the *Pairwise Oracle*. The results are shown in Table 2.

Results Although dynamic sampling was more effective for the pairwise tasks, we find that dynamic sampling was worse than sampling by size when using MTL on all nine datasets (top half of Table 2).

However, when the MTL_{All} method is compared to the pairwise methods, it does not perform as well (bottom half of Table 2). We see that the Pairwise Oracle, which uses the best supplementary task for the given target task, outperforms all methods by a large margin. Thus, although MTL_{All} is conceptually simple, it is not the best choice with respect to target task accuracy. Furthermore, if you could predict which supplementary task would be most effective (Pairwise Oracle, c.f. Section 4, Vu et al. (2020); Poth et al. (2021), etc.), you would be able to make even larger gains over MTL_{All}.

C Theories for Transfer Effectiveness

Previous work often invokes ideas such as catastrophic forgetting to describe why STILTs or MTL does or does not improve over the basic fine-tuning case (Phang et al., 2018; Pruksachatkun et al., 2020b; Wang et al., 2018a). However, as our work provides a novel comparison of MTL vs. STILTs there exists no previous work that shows how these methods differ in any practical or theoretical terms (e.g. does MTL or STILTs cause more catastrophic forgetting of the target task). Furthermore, previous explanations for why the STILTs method works has been called into question (Chang and Lu, 2021), leaving it an open research area.

A naive explanation for our task would be to think that when the target task is larger, STILTs should be worse because of catastrophic forgetting, whereas MTL would still have access to the supporting task. However, for STILTs this catastrophic forgetting would mainly effect the supporting task performance, not the target task performance, making that explanation unlikely in some contexts (e.g. when the tasks are not closely related). One potential explanation based on our results is that a small supporting task is best used to provide a good initialization for a larger target task (e.g. STILTs) while a large supporting task used for initialization would change the weights too much for the small target task to use effectively (thus making MTL the more effective strategy for a larger supporting task). Another explanation could be that a larger target task does not benefit from MTL (and perhaps is harmed by it, e.g. catastrophic interference) and therefore, STILTs is more effective - while MTL is more effective for small target tasks. However, all of these explanations also fail to take into account task relatedness, which likely also plays a role in the theoretical explanation (although even that too, has been called into question with Chang and Lu (2021)).

We thus note that there are a myriad of possible explanations (and the answer is likely a complex combination of possible explanations), but these are out of the scope of this work. Our work aims to show what happens in practice, rather than proposing a theoretical framework. As theoretical explanations for transfer learning are still an active area of research, we leave them to future work and provide this empirical comparison to guide their efforts and the current efforts of NLP researchers and practitioners.

D Alternate Model: BERT

We conduct the same analysis as Figure 1 with the BERT model and find similar results (Figure 3, thus showing that our results transfer to other pre-trained transformer models. We follow previous work in using two different pre-trained models for our analysis (Talmor and Berant, 2019; Phang et al., 2018).

E Additional Background Discussion

In this section we will show how the size heuristic is supported by and helps explain the results of previous work in this area. **Although this section is not crucial to the main result of our work, we include it to help readers who may not be as familiar with the related work**. We examine two works in depth and then discuss broader themes of related work.

BERT on STILTs Phang et al. (2018) This work defined the acronym STILTs, or *Supplementary Training on Intermediate Labeled-data Tasks*, which has been an influential idea in the community (Voskarides et al., 2019; Yan et al., 2020; Clark et al., 2020). To determine the effect of the intermediate training, the authors computed the STILTs matrix of each pair in the GLUE dataset. As our model and training framework are different from their methodology, we cannot compare our matrix with the absolute numbers in their matrix. However, at the end of Section 4 in their paper, they conduct an experiment with MTL and compare the results to their STILTs matrix (their experimental results are reproduced in Table 3 for convenience). Their analysis uses MNLI as the supporting task and RTE as the target task, trying MTL, STILTs, MTL+fine-tuning, and only fine-tuning on RTE. Their results show that STILTs provides the highest score, with all MTL varieties being worse. From this they conclude that MTL is worse than STILTs.

Model	RTE accuracy
GPT → RTE	54.2
GPT → MNLI → RTE	**70.4**
GPT → {MNLI, RTE}	68.6
GPT → {MNLI, RTE} → RTE	67.5

Table 3: Table reproduced from Phang et al. (2018). Their comparison of STILTs against MTL setups for GPT, with MNLI as the intermediate task and RTE as the target task. Only one run was reported (e.g. no standard error or confidence intervals).

How does this compare to our results? In Figure 1 we see that our results also show that the STILTs method is better than the MTL method for the (RTE, MNLI) pair, showing that our results are consistent with those in the literature. Furthermore, we find that this is one of the 4 significant cells in our matrix where the size heuristic does not accurately predict the best method. It is unfortunate that the task they decided to pick happened to be one of the anomalies. Thus, our paper extends and completes their results with more rigor.

MultiQA Talmor and Berant (2019) MultiQA showed that using MTL on a variety of question-answering (QA) datasets made it possible to train a model that could outperform the current SOTA on those QA datasets. They used an interesting approach to MTL, pulling 15k examples from each of the 5 major datasets to compose one new “MTL" task, called Multi-75K. They then show results for STILTs transfer on those same datasets along with the MTL dataset (their data is reproduced with new emphasis in Appendix E Table 4 for conve-

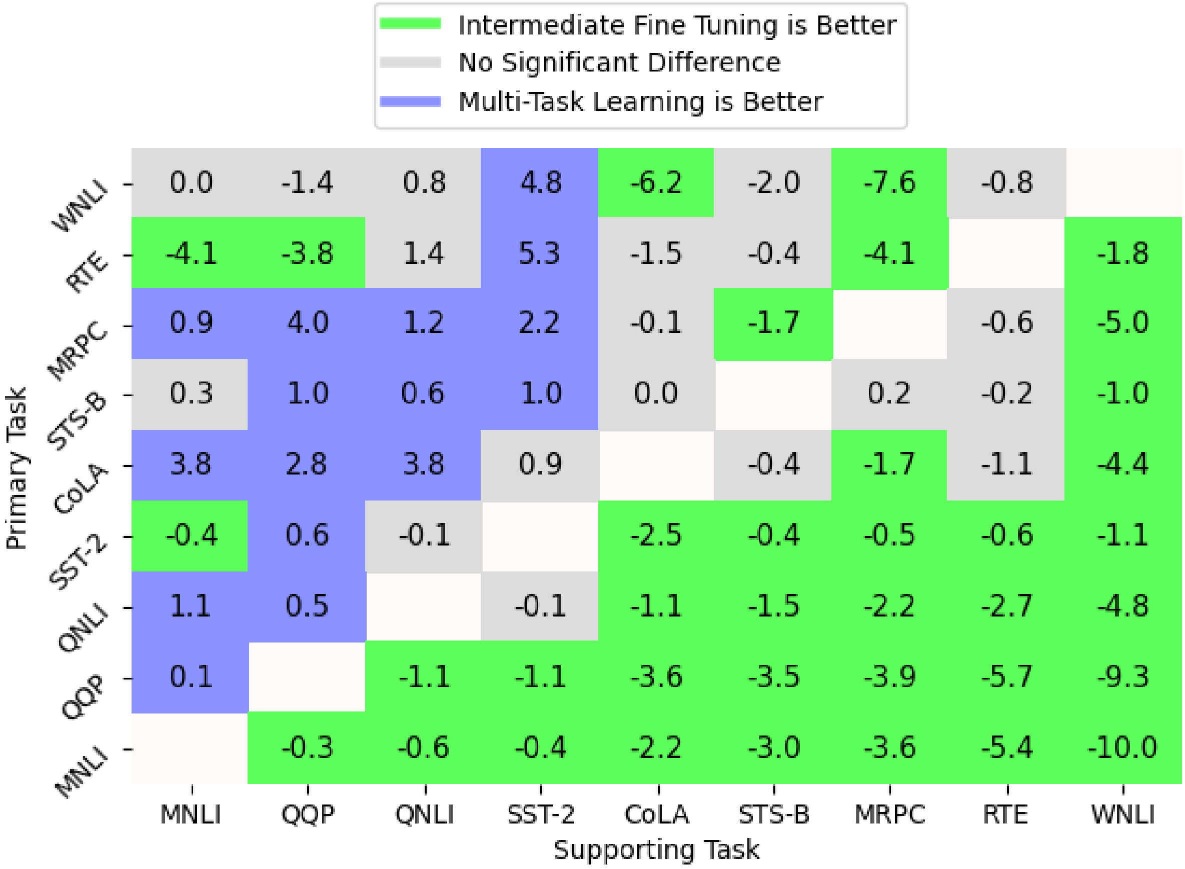


Figure 3: Results comparing intermediate fine tuning (STILTs) vs multi-task learning (MTL) with the BERT model. Numbers in cells indicate the absolute percent score difference on the primary task when using MTL instead of STILTs (positive scores mean MTL is better and vice versa). The colors indicate visually the best method, showing a statistically significant difference from the other from using using a two-sided t-test with $\alpha = 0.1$. Datasets are ordered in descending size.

nience). We note that this STILTs-like transfer with the “MTL" dataset is an equivalent method to doing MTL and then fine-tuning on the target task, reminiscent of the third example in Phang et al. (2018) (Table 3, GPT → {MNLI, RTE} → RTE, c.f. Appendix E).

How does this relate to our results? The size heuristic says that MTL is better than STILTs when the target task has fewer training instances. In the MultiQA paper the size of each training set is artificially controlled to be the same number (75k instances), thus our size heuristic would say that the methods should be comparable. Although no error bounds or standard deviations are reported in their paper (which makes the exact comparison difficult), we see that the MTL approach performs equal or better on almost half of the datasets. Thus, although the MultiQA paper is not strictly comparable to our work due to their training setup (the MTL+fine tuning), their results agree with our hypothesis as well.

For convenience, Table 4 from Talmor and Berant (2019) is reproduced here in the appendix. The top half contains the results using the DocQA model while the bottom half uses BERT. Note that both model’s Multi-75K scores perform approximately similar to the STILTs methods, which is expected given that they are the same size. TQA-G and TQA-W come from the same dataset. As stated in the body of this paper, no standard deviation is reported in the MultiQA paper and thus it is hard to know whether the difference in results are statistically significant. Even if all results were statistically significant, which is highly unlikely, each of the Multi-75K models perform equal or better on 2 of the 6 tasks, which is not statistically different from random.

Combining All Tasks Our results using MTL_{All} showed that although MTL_{All} is conceptually easy (just put all the datasets together) it does not lead to the best performance. We find similar results in Wang et al. (2018a), where in their Table 3 they show that the STILTs approach outperforms the

	SQuAD	NewsQA	SearchQA	TQA-G	TQA-W	HotpotQA
SQuAD	-	**33.3**	39.2	49.2	34.5	17.8
NewsQA	59.6	-	41.6	44.2	33.9	16.5
SearchQA	57	31.4	-	**57.5**	39.6	**19.2**
TQA-G	57.7	31.8	**49.5**	-	**41.4**	19.1
TQA-W	57.6	31.7	44.4	50.7	-	17.2
HotpotQA	**59.8**	32.4	46.3	54.6	37.4	-
Multi-75K	**59.8**	33.0	47.5	56.4	40.4	**19.2**
SQuAD	-	41.2	47.8	55.2	45.4	20.8
NewsQA	**72.1**	-	47.4	55.9	45.2	20.6
SearchQA	70.2	40.2	-	**57.3**	45.5	20.4
TQA-G	69.9	41.2	**50.0**	-	46.2	20.8
TQA-W	71.0	39.2	48.4	55.7	-	**20.9**
HotpotQA	71.2	39.5	48.6	56.6	45.6	-
Multi-75K	71.5	**42.1**	48.5	56.6	**46.5**	20.4

Table 4: Results taken from the right half of Table 4 in the MultiQA paper (Talmor and Berant, 2019) as that section is directly relevant to this work (the *self* row containing only standard fine-tuning is removed for clarity). Emphasis changed to reflect the best score in the model's column instead of the best non-MTL score.

MTL$_{\text{All}}$ approach for all but one task. Additionally, in the follow up work from the initial STILTs paper (Phang et al., 2020) they find that although MTL$_{\text{All}}$ has a slightly higher average performance in the cross-lingual setting, it is worse than the pairwise approach in 75% of the evaluated tasks.

The current literature (and our work) seems to suggest that naively combining as many tasks as possible may not be the best approach. However, more work is needed to understand the training dynamics of MTL$_{\text{All}}$.

Combining Helpful Tasks In this paper, we only examine the difference between pairwise MTL, STILTs or MTL$_{\text{All}}$, due to time and space. Although it is possible that our heuristic may extrapolate to transfer learning with more than two tasks, computing the power set of the possible task combinations for MTL and STILTs would be extremely time and resource intensive. We leave it to future work to examine how the size heuristic may hold when using more than two datasets at a time.

Additionally, there may be further value in computing this power set: Changpinyo et al. (2018) showed that taking the pairwise tasks that proved beneficial in pairwise MTL and combining them into a larger MTL set (an "Oracle" set) oftentimes provides higher scores than pairwise MTL. Exploring which subsets of tasks provide the best transfer with which method would be valuable future work.

Dataset Size in TL Dataset size has been used often in transfer learning techniques (Søgaard and Bingel, 2017; Pruksachatkun et al., 2020a; Poth et al., 2021). Our size heuristic, although related, focuses on a different problem: whether to use MTL or STILTs. Thus, our work provides additional insight into how the size of the dataset is important for transfer learning.

Fine-tuning after MTL Many papers that use MTL$_{\text{All}}$ also perform some sort of fine-tuning after the MTL phase. Since fine-tuning after MTL makes the MTL phase an intermediate step, it essential combines the STILTs and MTL methods into a single STILTs-like method. However, whether fine-tuning after MTL is better than simply MTL is still controversial: for example, Liu et al. (2019b), Raffel et al. (2019), and Talmor and Berant (2019) say that fine-tuning after MTL helps but Lourie et al. (2021) and Phang et al. (2018) say that it doesn't. However, Raffel et al. (2019) is the only one whose experiments include multiple random seeds, giving more credence to their results. However, due to the difference of opinion it is unclear which method is actually better; we leave this to future work.

F GLUE Dataset Sizes and References

To give credit to the original authors and to provide the exact sizes, we provide Table 5.

Dataset	Citation	Training Size
MNLI	Williams et al. (2018)	392,662
QQP	No citation, link here	363,846
QNLI	Levesque et al. (2011)	104,743
SST-2	Socher et al. (2013)	67,349
CoLA	Warstadt et al. (2018)	8,551
STS-B	Cer et al. (2017)	5,749
MRPC	Dolan and Brockett (2005)	3,668
RTE	Dagan et al. (2006)*	2,490
WNLI	Levesque et al. (2011)	635

Table 5: Sizes of the datasets in GLUE (Wang et al., 2018b) in descending order, along with their original citations. RTE is compiled from these sources: Dagan et al. (2006); Bar Haim et al. (2006); Giampiccolo et al. (2007); Bentivogli et al. (2009)

Leveraging Explicit Lexico-logical Alignments in Text-to-SQL Parsing

Runxin Sun[1,2], **Shizhu He**[1,2], **Chong Zhu**[1,2], **Yaohan He**[3], **Jinlong Li**[3],
Jun Zhao[1,2] and **Kang Liu**[1,2,4]
[1] National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3] AI Lab, China Merchant Bank, ShenZhen, 518057, China
[4] Beijing Academy of Artificial Intelligence, Beijing, 100084, China
sunrunxin2020@ia.ac.cn, {shizhu.he, chong.zhu}@nlpr.ia.ac.cn,
{heyh18, lucida}@cmbchina.com, {jzhao, kliu}@nlpr.ia.ac.cn

Abstract

Text-to-SQL aims to parse natural language questions into SQL queries, which is valuable in providing an easy interface to access large databases. Previous work has observed that leveraging lexico-logical alignments is very helpful to improve parsing performance. However, current attention-based approaches can only model such alignments at the token level and have unsatisfactory generalization capability. In this paper, we propose a new approach to leveraging explicit lexico-logical alignments. It first identifies possible phrase-level alignments and injects them as additional contexts to guide the parsing procedure. Experimental results on SQUALL show that our approach can make better use of such alignments and obtains an absolute improvement of 3.4% compared with the current state-of-the-art.

1 Introduction

Text-to-SQL parsing is the task of mapping natural language questions to executable SQL queries on relational databases (Zhong et al., 2017). It provides an easy way for common users unfamiliar with query languages to access large databases and has attracted great attention. Recently, *lexico-logical* alignments, which align question phrases to their corresponding SQL query fragments, have been proved to be very helpful in improving parsing performance (Shi et al., 2020). As shown in Figure 1, the token "competitor" should be aligned to "c1" in the SQL query. To capture such alignments, several attention-based models were proposed (Shi et al., 2020; Lei et al., 2020; Liu et al., 2021), which employ the attention weights among tokens to indicate the alignments. Specifically, they use an attention module to perform schema linking at the encoding stage (Lei et al., 2020; Liu et al., 2021), and may use another attention to align each output token to its corresponding input tokens at the decoding stage (Shi et al., 2020).

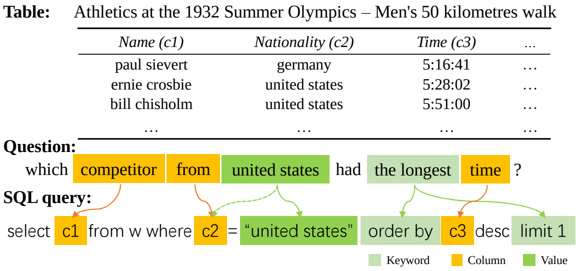


Table:
Athletics at the 1932 Summer Olympics – Men's 50 kilometres walk

Name (c1)	Nationality (c2)	Time (c3)	...
paul sievert	germany	5:16:41	...
ernie crosbie	united states	5:28:02	...
bill chisholm	united states	5:51:00	...
...	...	...	...

Question:
which competitor from united states had the longest time ?
SQL query:
select c1 from w where c2 = "united states" order by c3 desc limit 1
Keyword
Column
Value

Figure 1: An example from SQUALL. Alignments belonging to the same type are marked with the same color.

However, we argue that the attention mechanism is not an appropriate way to capture and leverage lexico-logical alignments. It mainly has the following two problems. First, the standard attention can only model alignments at the token level rather than the phrase level, while there are many multi-granular, non-continuous alignments in the text-to-SQL task. For the example in Figure 1, "`order by ... limit 1`" is a SQL keyword pattern representing a superlative operation. However, the standard attention module can only align "`order`", "`by`", "`limit`", and "`1`" to "the longest" token by token, rather than regarding them as a whole. It may confuse the decoder and lead to the failure to generate this pattern correctly (Herzig and Berant, 2021). Second, traditional attention-based approaches are prone to overfitting the training data, which is harmful to the model's generalization capability. It is not only the *domain generalization* (Dong et al., 2019) but also the *compositional generalization* (Herzig and Berant, 2021). Specifically, the former refers to the generalization across different databases, while the latter refers to the ability to generate new structures composed of seen components.

To solve the aforementioned problems, we propose a neural parsing framework to leverage explicit lexico-logical alignments. Dong et al. (2019) have pointed out that if we align question tokens

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 283 - 289
May 22-27, 2022 ©2022 Association for Computational Linguistics

to columns or values in databases before parsing, it will help to improve the model's generalization among different domains (databases). Motivated by this, our framework consists of two steps. Specifically, we first implement a simple model to obtain possible lexico-logical alignments before parsing. While in the second step, we inject such alignments into a standard seq2seq parser by treating them as additional contexts, similar to "prompt information" or "evidence" in machine reading comprehension (Mihaylov and Frank, 2018; Tu et al., 2020; Niu et al., 2020). Moreover, to alleviate the negative effects on the parser caused by noise alignments, we propose a data augmentation method that adds noisy alignments during the training procedure. Experimental results on an open-released dataset, SQUALL (Shi et al., 2020), show that our framework achieves state-of-the-art performance and obtains an absolute improvement of 3.4% compared with existing attention-based models.

2 Preliminaries

2.1 Problem Definition

Here we consider the problem setting adopted by Shi et al. (2020). Formally, given a natural language question Q about a table T, our goal is to generate the corresponding SQL query Y, where the table consists of columns $\{c_1, \dots, c_{|T|}\}$.

2.2 Base Parser

Our base parser is a standard seq2seq model. It generally follows the architecture proposed by Lin et al. (2020), which combines a BERT-based encoder with a sequential pointer-generator to perform an end-to-end parsing procedure.

Input Serialization and Encoder According to the definition above, an input X contains a length-n question $Q = q_1, \dots, q_n$ and a table with m columns $T = \{c_1, \dots, c_m\}$. We concatenate all the columns into a sequence for the table, where a unique token precedes each column to represent its type (e.g., text). Then we add two `[SEP]` tokens at both ends and append this sequence to the question. After adding a `[CLS]` token at the beginning, we get the input sequence in the following format:

$$X = \texttt{[CLS]}, Q, \texttt{[SEP]}, \texttt{[TYPE\#C1]}, c_1, \\ \dots, \texttt{[TYPE\#Cm]}, c_m, \texttt{[SEP]}$$

X is encoded with BERT (Devlin et al., 2019), followed by a bidirectional LSTM (bi-LSTM) to get the hidden representations $\boldsymbol{h}_X$. Then for the question part, we feed its representation to another bi-LSTM to obtain the encoding result $\boldsymbol{h}_Q$. Each column is represented by the vector of its corresponding type token.

Decoder Like Lin et al. (2020), We use an LSTM-based pointer-generator (See et al., 2017) enhanced with the attention mechanism as the decoder. Specifically, we use the final hidden state of the question encoder to initialize the decoder. At each step t, the decoder chooses one of the following three actions: generating a keyword from the vocabulary V, copying a token from the question Q, or copying a column from the table T.

3 Method

3.1 Framework Overview

As shown in Figure 2, our framework consists of two stages: *lexico-logical alignment prediction* (the upper left) and *alignment-enhanced parsing* (the bottom). At the first stage (*alignment prediction*), we identify possible lexico-logical alignments in the question before parsing. At the second stage (*alignment-enhanced parsing*), we inject these alignments into the parser so that it can make further completions and refinements based on them.

3.2 Lexico-logical Alignment Prediction

In this step, we implement a simple model to predict lexico-logical alignments of the input question. Specifically, we adopt a two-stage pipeline process: 1) identify question phrases that may have alignments; 2) predict their corresponding query fragments according to the types.

For the first stage, we classify the alignments into three types according to their corresponding query fragments: *keyword*, *column*, *value*. Specifically, ***keyword*** alignments map question phrases to query fragments composed of SQL keywords, while the other two types of alignments (***column*** and ***value***) map them to columns in databases. The only difference between ***column*** and ***value*** alignments is that the phrase part of a *value* alignment is also a value in the SQL query. Analogous to Named Entity Recognition (NER), we use sequence labeling to implement this process:

$$P(label_i \mid Q, T) = \text{softmax}(\text{MLP}([\boldsymbol{h}_i; \boldsymbol{c}_i])). \tag{1}$$

Here we apply the BIO labeling schema, classifying each token as one of the four types: *keyword*,

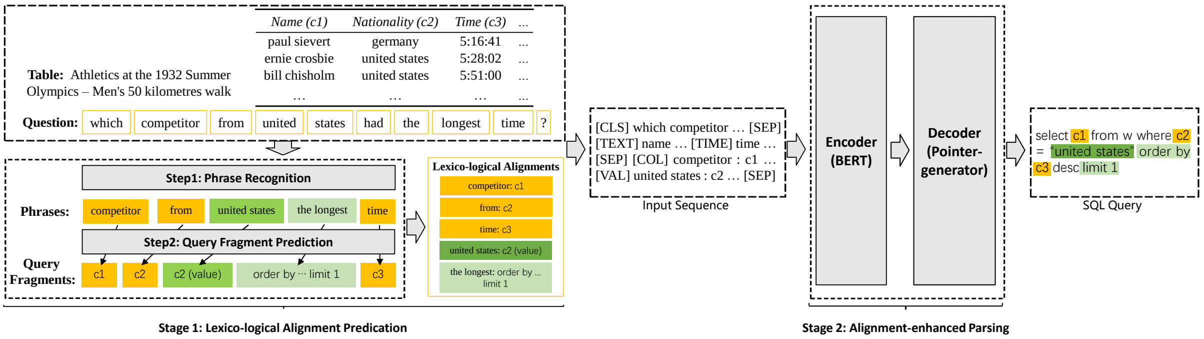

Name (c1)
Nationality (c2)
Time (c3)
...
paul sievert
germany
5:16:41
ernie crosbie
united states
5:28:02
bill chisholm
united states
5:51:00
Table: Athletics at the 1932 Summer Olympics – Men's 50 kilometres walk
Question:
which
competitor
from
united
states
had
the
longest
time
?
Step1: Phrase Recognition
Phrases:
competitor
from
united states
the longest
time
Step2: Query Fragment Prediction
Query Fragments:
c1
c2
c2 (value)
order by ··· limit 1
c3
Lexico-logical Alignments
competitor: c1
from: c2
time: c3
united states: c2 (value)
the longest: order by ... limit 1
Stage 1: Lexico-logical Alignment Predication
[CLS] which competitor … [SEP]
[TEXT] name … [TIME] time …
[SEP] [COL] competitor : c1 …
[VAL] united states : c2 … [SEP]
Input Sequence
Encoder (BERT)
Decoder (Pointer-generator)
select c1 from w where c2 = "united states" order by c3 desc limit 1
SQL Query
Stage 2: Alignment-enhanced Parsing

Figure 2: An illustration of our framework. It consists of two stages: (i) lexico-logical alignment prediction; (ii) alignment-enhanced parsing.

column, *value*, or *none*. We adopt the same structure as our base parser to encode the input sequence, and $\boldsymbol{h}_i$ is the hidden representation of the i-th token. Moreover, an attention module is used to get the column-aware question representation $\boldsymbol{c}_i$:

$$\boldsymbol{c}_i = \text{Attention}(\boldsymbol{h}_i, \boldsymbol{h}_C, \boldsymbol{h}_C), \qquad (2)$$

where $\boldsymbol{h}_C$ are the representations of all the columns. Then we run a Multi-Layer Perceptron (MLP) by concatenating these two vectors as inputs to predict the i-th label.

For the second stage, we predict the query fragment corresponding to the phrase. Specifically, we can divide this process into the following two cases according to the type of phrase:

1) *Keyword*: We use a generation model to obtain keyword fragments corresponding to such phrases. In detail, we perform self-attention on the token representations of the phrase p to get the initial hidden state. Then we run an RNN model with attention to generate its corresponding keyword fragment:

$$y = \arg\max \prod_t P(y_t \mid y_{<t}, p). \qquad (3)$$

2) *Column* & *Value*: In this case, we should link the phrase to its corresponding column. Intuitively, based on the attention matrix, we can directly get the column c^* that best matches the phrase p:

$$c^* = \arg\max_{c \in C} f(p, c) = \arg\max_{c \in C} \sum_{w \in p} f(w, c). \qquad (4)$$

3.3 Alignment-enhanced Parsing

After getting all the lexico-logical alignments in a question, we then consider adding them to the parsing process. Naturally, we design their usages for both the encoding and decoding processes.

For the encoding stage, we treat alignments as additional contexts and add them to the input sequence. Concretely, we represent each alignment as a concatenation of the natural language phrase p and its corresponding query fragment f, where the two parts are separated by ":". Moreover, a unique token before each of them represents its type (*keyword*, *column* or *value*). Thus the format of the modified input sequence is as follows:

$$\begin{aligned} X^+ = & \texttt{[CLS]}, Q, \texttt{[SEP]}, \texttt{[TYPE\#C1]}, c_1, \ldots, \\ & \texttt{[TYPE\#Cm]}, c_m, \texttt{[SEP]}, \texttt{[TYPE\#A1]}, \\ & p_1, :, f_1, \ldots, \texttt{[TYPE\#An]}, p_n, :, f_n \texttt{[SEP]} \end{aligned}$$

In this way, the encoder based on a pre-trained language model can make good use of this information to help it better perform schema linking.

For the decoding stage, we also add alignments to the generation process. Specifically, we take its type token's hidden vector as the representation of each alignment, denoting it as $\boldsymbol{h}_A$. So at each step t, we compute the attention between the decoder hidden state $\boldsymbol{h}_t^D$ and the alignments:

$$\boldsymbol{a}_t = \text{Attention}(\boldsymbol{h}_t^D, \boldsymbol{h}_A, \boldsymbol{h}_A). \qquad (5)$$

Afterward, we use the concatenation of $\boldsymbol{a}_t$ and the embedding of the previous token $\boldsymbol{e}_t$ as the decoder's input, injecting this information into the next step's hidden state:

$$\boldsymbol{h}_{t+1}^D = \text{LSTM}^D([\boldsymbol{e}_t; \boldsymbol{a}_t], \boldsymbol{h}_t^D). \qquad (6)$$

3.4 Noisy Alignment Augmentation

As mentioned before, it is impossible to obtain a perfect model for alignment prediction. So if we use the annotated alignments to train the parser, and

Model	Dev		Test
	ACC$_{\text{LF}}$	ACC$_{\text{EXE}}$	ACC$_{\text{EXE}}$
SEQ2SEQ$^+$ + BERT	44.7 ± 2.1	63.8 ± 1.1	51.8 ± 0.4
ALIGN + BERT	47.2 ± 1.2	66.5 ± 1.2	54.1 ± 0.2
LAP (ours)	47.0 ± 1.3	65.0 ± 1.2	53.0 ± 0.5
+ noisy alignment	**50.6 ± 1.0**	**68.3 ± 0.8**	**56.5 ± 0.3**

Table 1: Overall parsing results. LAP refers to our model. "+ noisy alignment" means our model training under the noisy alignment augmentation.

use the predicted alignments to make predictions, then there is an inconsistency between training and testing. It is precisely because of this inconsistency that the parser tends to trust the given alignments completely. In that case, wrong alignments may hurt the parsing performance.

To alleviate the negative effects on the parser caused by noise alignments, we propose a method based on data augmentation, that is, adding noisy alignments during the training procedure. Specifically, we use the model proposed in section 3.2 to predict alignments for the training examples through cross-validation. Obviously, these alignments are noisy. Then we integrate these predicted examples with the annotated examples and use them as the augmented training set of the parser.

4 Experiments

4.1 Dataset and Experimental Setup

We evaluate on SQUALL (Shi et al., 2020), a large-scale dataset based on WIKITABLEQUESTIONS (Pasupat and Liang, 2015). It contains 11,276 table-question-answer triplets, enriched with human-annotated logical forms and lexical-logical alignments.[1] We use the default dataset split provided by Shi et al. (2020), where they randomly shuffle the tables and divide them into five splits so that examples with the same table are in the same split.

For evaluation metrics, we employ the average logical form accuracy ACC$_{\text{LF}}$ and execution accuracy ACC$_{\text{EXE}}$,[2] following Shi et al. (2020). For model implementation, please refer to Appendix A for more details. It is worth noting that, unless otherwise stated, we only use the alignment annotations of the training set to train the alignment prediction model. While on the dev / test set, we use the predicted alignments as the parser's input.

Model	DB split	Query split	IID split
SEQ2SEQ + BERT	43.5	1.2	48.1
+ attention sup.	46.7 (+ 3.2)	1.6 (+ 0.4)	51.1 (+ **3.0**)
LAP (w/o alignment)	46.6	2.7	52.1
+ noisy alignment	50.0 (+ **3.4**)	3.5 (+ **0.8**)	53.0 (+ 0.9)

Table 2: Parsing results (ACC$_{\text{LF}}$) over different splits of SQUALL. "+ attention sup." refers to using alignment annotations to supervise the attention module. LAP (w/o alignment) refers to our model without alignments.[3]

Model	ACC$_{\text{LF}}$ (Dev)	Δ
SEQ2SEQ + BERT	43.5	
+ oracle attention	66.3	+ 22.8
LAP (w/o alignment)	46.6	
+ keyword alignment	58.1	+ 11.5
+ column alignment	55.1	+ 8.5
+ value alignment	54.1	+ 7.5
+ oracle alignment (token)	71.9	+ 25.3
+ oracle alignment	**73.1**	**+ 26.5**

Table 3: Parsing results on the 0-split under the oracle setting. SEQ2SEQ + BERT refers to the base parser (Shi et al., 2020) with BERT embeddings.

4.2 End-to-end Parsing Performance

To evaluate the effectiveness of our model, we compare end-to-end parsing performance with existing attention-based models. The results are shown in Table 1. For the baselines, we select SEQ2SEQ$^+$ and ALIGN provided by Shi et al. (2020). The former uses the automatically derived exact-match features to supervise the attention modules, while the latter uses the alignment annotations instead.

From the results, we can observe that after combining the alignment prediction model proposed in section 3.2, our parser (LAP) achieves state-of-the-art performance on SQUALL. We believe the reason is that our approach identifies possible lexico-logical alignments before parsing so that the parser can leverage such explicit alignments and model them on the phrase level. Moreover, "LAP + noisy alignment" further outperforms "LAP". It illustrates that noise alignments do have negative effects on the parser, while our noisy alignment augmentation method can alleviate them effectively.

4.3 Our Model's Generalization Capability

To evaluate the advantages of our model's generalization capability, we further made different splits of SQUALL (Shi et al., 2020) and conducted experiments on them. Here we evaluate the model's generalization capability from two perspectives: *domain generalization* and *compositional generalization*. Specifically, *DB split* refers to the default

[1] There are no such annotations for the test set.

[2] ACC$_{\text{LF}}$ checks whether the logical form output exactly matches the target, while ACC$_{\text{EXE}}$ compares the execution results.

[3] Please refer to Appendix B for more details.

cross-DB setting of SQUALL, where databases appearing in the test set were not seen during training, and we use it to test the model's *domain generalization*. *Query split* is the setting proposed by Finegan-Dollak et al. (2018) to test the model's *compositional generalization*, where no query template (query after anonymization of database-related variables) appears in more than one set. As for the *IID split*, it means that the test case is not in the training set while its corresponding database is seen during training. We employ it as the control group.[4]

The experimental results are shown in Table 2. From the results, we can observe that our approach (+ noisy alignment) obtains more significant improvement on the DB split and the query split, while it is not as effective as the attention-based approach on the IID split. In particular, our approach achieves twice the improvement on the query split (0.8 vs. 0.4), even on a stronger base parser. These reveal that our approach is more effective when parsing across different databases (*domain generalization*) and different query templates (*compositional generalization*), which illustrates that our approach has better generalization capability.

4.4 The Effectiveness of our Parser on Leveraging Lexico-logical Alignments

To evaluate the effectiveness of our parser on the lexico-logical alignments utilization, we conducted experiments under the oracle setting, where we used alignment annotations instead of predictions for testing. Table 3 shows the results.[5]

From the results, we can observe that 1) our parser obtains more improvements when injecting alignments (+ oracle alignment) than the attention-based approach (+ oracle attention). It proves that our model could more effectively utilize the *lexico-logical* alignment information. 2) We also show the results when injecting different types of alignment in our model. The results show that keyword alignment, which is excluded from traditional schema linking, is a valuable type and is also helpful in improving parsing performance. 3) Modeling such alignments at the phrase-level is more effective than the token-level ("+ oracle alignment" vs. "+ oracle alignment (token)").

5 Conclusion

In this paper, we propose a neural parsing framework to leverage explicit lexico-logical alignments by treating them as additional contexts. Moreover, to alleviate the negative effects on the parser caused by noise alignments, we add noisy alignments during training inspired by data augmentation. Experimental results on SQUALL show that our framework achieves state-of-the-art performance compared with existing attention-based models.

Acknowledgments

This work is supported by the Natural Key R&D Program of China (No.2020AAA0106400), the National Natural Science Foundation of China (No.61922085, No.61976211) and the Key Research Program of the Chinese Academy of Sciences (Grant NO.ZDBS-SSW-JSC006). This research work was supported by the independent research project of National Laboratory of Pattern Recognition, the Youth Innovation Promotion Association CAS and Yunnan Provincial Major Science and Technology Special Plan Projects (No.202103AA080015).

References

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

Zhen Dong, Shizhao Sun, Hongzhi Liu, Jian-Guang Lou, and Dongmei Zhang. 2019. Data-anonymous encoding for text-to-SQL generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5405–5414, Hong Kong, China. Association for Computational Linguistics.

Catherine Finegan-Dollak, Jonathan K Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia. Association for Computational Linguistics.

Jonathan Herzig and Jonathan Berant. 2021. Span-based semantic parsing for compositional general-

[4] Please refer to Appendix B for more details.

[5] Because Shi et al. (2020) did not provide the oracle results with BERT, we re-ran the open-source code (`https://github.com/tzshi/squall`) and got the results. Besides, due to the limitation of resources, we conducted them only on the 0-split of SQUALL instead of all five splits.

ization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, page 908–921, Online. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Wenqiang Lei, Weixin Wang, Zhixin Ma, Tian Gan, Wei Lu, Min-Yen Kan, and Tat-Seng Chua. 2020. Re-examining the role of schema linking in text-to-SQL. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6943–6954, Online. Association for Computational Linguistics.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2020. Bridging textual and tabular data for cross-domain text-to-SQL semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4870–4888, Online. Association for Computational Linguistics.

Qian Liu, Dejian Yang, Jiahui Zhang, Jiaqi Guo, Bin Zhou, and Jian-Guang Lou. 2021. Awakening latent grounding from pretrained language models for semantic parsing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1174–1189, Online. Association for Computational Linguistics.

Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 821–832, Melbourne, Australia. Association for Computational Linguistics.

Yilin Niu, Fangkai Jiao, Mantong Zhou, Ting Yao, Jingfang Xu, and Minlie Huang. 2020. A self-training method for machine reading comprehension with soft evidence extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3916–3927, Online. Association for Computational Linguistics.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: an imperative style, high-performance deep learning library. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 8026–8037, Red Hook, NY, USA. Curran Associates Inc.

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

Tianze Shi, Chen Zhao, Jordan Boyd-Graber, Hal Daumé III, and Lillian Lee. 2020. On the potential of lexico-logical alignments for semantic parsing to SQL queries. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1849–1864, Online. Association for Computational Linguistics.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9073–9080.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

A Model Implementation Details

Our model is implemented in PyTorch (Paszke et al., 2019). For the BERT model, we fine-tune a `bert-base-uncased` model from the *Hugging Face's Transformers* library (Wolf et al., 2020). For the attention module, we use the standard dot-product attention function. We set all LSTMs to 1-layer and hidden size to 256. We use the Adam optimizer (Kingma and Ba, 2015) and clip gradients to 2.0. For the loss function, we choose cross-entropy for the classification task and label-smoothing for the generation task. We train our alignment prediction model for up to 10 epochs and SQL parser for 20 epochs. Both of them have an epoch for warm-up, and then the learning rate will decay linearly.

In terms of hyperparameter search, we turned the batch size (**8**, 16, 32), max learning rate (1e-3, **1e-4**), max BERT learning rate (5e-5, **2e-5**, 1e-5, 5e-6), and dropout (0.1, **0.2**, 0.3, 0.5). Due to the limitation of resources, we turned these parameters one by one instead of using grid search. The bolded values are a set of optimal parameters we found.

B Details of Different Splits of the SQUALL Dataset

We made three different splits of the SQUALL dataset: *IID split*, *DB split*, and *query split*, to explore the corresponding generalization capabilities of the model. It is worth noting that because this dataset is a single-table dataset (that is, each DB contains only one table), the cross-DB setting is essentially equal to the cross-table setting. The specific methods for obtaining these splits are as follows:

- ***IID split***: In order to ensure that tables in the test set are also in the training set, and the only difference between the two sets is that the included samples are different, we classify the samples according to their corresponding tables. For each category (i.e., table), we randomly select k (in this case, $k = 1$) samples, put them into the test set, and put the rest into the training set.

- ***DB split***: This is the default setting of the SQUALL dataset. Here we use the 0-split provided by Shi et al. (2020).

- ***query split***: Inspired by Finegan-Dollak et al. (2018), we substitute variables for table-related entities (i.e., columns and values) in each query in the dataset to obtain its corresponding query template, just like Shi et al. (2020) did. Similarly, we classify the samples according to their corresponding query templates. For each category (i.e., query template), all its samples can only be put into either the training set or the test set. It is worth noting that to examine the *compositional generalization* better, we sort the templates according to their frequency. Then we put the templates with higher frequency into the training set and the templates with lower frequency into the test set.

For the above three splits, we make the ratio of the training set and the test set approximately equal to 4:1, consistent with Shi et al. (2020).

C Details on Obtaining Token-level Alignments

To verify whether our approach can model alignments at the phrase level, we constructed token-level alignments to contrast with the original alignment annotations. Specifically, we imitated the attention mechanism and decomposed the alignments according to their types.

For *keyword* alignments, inspired by Shi et al. (2020), we align each keyword in the SQL query to all its corresponding tokens in the question. For the example in Figure 1, we align "`order`", "`by`", "`limit`", and "`1`" to "the longest" respectively. Then we obtain the following four alignments: "the longest: `order`", "the longest: `by`", "the longest: `limit`", and "the longest: `1`".

For the other two types of alignments: *column* and *value*, as mentioned in section 3.2, they both align question phrases to columns in databases. Analogous to schema linking through an attention module, we align each token in the question phrase to the corresponding column separately. For the example in Figure 1, we align "united" and "states" to column `c2` respectively instead of treating these two tokens as a whole.

Complex Evolutional Pattern Learning for Temporal Knowledge Graph Reasoning

Zixuan Li[1,2,3]*, Saiping Guan[1,2], Xiaolong Jin[1,2], Weihua Peng[3], Yajuan Lyu[3], Yong Zhu[3], Long Bai[1,2], Wei Li[3], Jiafeng Guo[1,2], Xueqi Cheng[1,2]

[1]School of Computer Science and Technology, University of Chinese Academy of Sciences;
[2]CAS Key Laboratory of Network Data Science and Technology,
Institute of Computing Technology, Chinese Academy of Sciences; [3]Baidu Inc.
{lizixuan,guansaiping,jinxiaolong}@ict.ac.cn
{pengweihua,lvyajuan,zhuyong}@baidu.com

Abstract

A Temporal Knowledge Graph (TKG) is a sequence of KGs corresponding to different timestamps. TKG reasoning aims to predict potential facts in the future given the historical KG sequences. One key of this task is to mine and understand evolutional patterns of facts from these sequences. The evolutional patterns are complex in two aspects, length-diversity and time-variability. Existing models for TKG reasoning focus on modeling fact sequences of a fixed length, which cannot discover complex evolutional patterns that vary in length. Furthermore, these models are all trained offline, which cannot well adapt to the changes of evolutional patterns from then on. Thus, we propose a new model, called Complex Evolutional Network (CEN), which uses a length-aware Convolutional Neural Network (CNN) to handle evolutional patterns of different lengths via an easy-to-difficult curriculum learning strategy. Besides, we propose to learn the model under the online setting so that it can adapt to the changes of evolutional patterns over time. Extensive experiments demonstrate that CEN obtains substantial performance improvement under both the traditional offline and the proposed online settings.

1 Introduction

Temporal Knowledge Graph (TKG) (Boschee et al., 2015; Gottschalk and Demidova, 2018, 2019; Zhao, 2020) has emerged as a very active research area over the last few years. Each fact in TKGs is a quadruple *(subject, relation, object, timestamp)*. A TKG can be denoted as a sequence of KGs with timestamps, each of which contains all facts at the corresponding timestamp. TKG reasoning aims to answer queries about future facts, such as *(COVID-19, New medical case occur, ?, 2022-1-9)*.

To predict future facts, one challenge is to dive deep into the related historical facts, which reflect the preferences of the related entities and affect their future behaviors to a certain degree. Such facts, usually temporally adjacent, may carry informative sequential patterns, called evolutional patterns in this paper. For example, [*(COVID-19, Infect, A, 2021-12-21), (A, Discuss with, B, 2021-12-25), (B, Go to, Shop, 2021-12-28)*] is an informative evolutional pattern for the above query implied in historical KGs. There are two kinds of models to model evolutional patterns, namely, query-specific and entire graph based models. The first kind of models (Jin et al., 2020; Li et al., 2021a; Sun et al., 2021; Han et al., 2020a, 2021; Zhu et al., 2021) extract useful structures (i.e., paths or subgraphs) for each individual query from the historical KG sequence and further predict the future facts by mining evolutional patterns from these structures. This kind of models may inevitably neglect some useful evolutional patterns. Therefore, the entire graph based models (Deng et al., 2020; Li et al., 2021a) take a sequence of entire KGs as the input and encode evolutional patterns among them, which exhibit superiority to the query-specific models.

However, they all ignore the length-diversity and time-variability of evolutional patterns. **Length-diversity**: The lengths of evolutional patterns are diverse. For example, [*(COVID-19, Infect, A, 2021-12-21), (A, Discuss with, B, 2021-12-25), (B, Go to, Shop, 2021-12-28)*] is a useful evolutional pattern of length 3 to predict the query *(COVID-19, New medical case occur, ?, 2022-1-9)* and [*(COVID-19, Infect, A, 2021-12-21), (A, Go to, Shop, 2021-12-30)*] is also a useful evolutional pattern of length 2 for this query. Previous models extract evolutional patterns of a fixed length, which cannot handle evolutional patterns of diverse lengths. **Time-variability**: Evolutional patterns change over time. For example, *(COVID-19, Infect, A, 2019-12-9)* and *(COVID-19, Infect, A, 2022-1-9)* may lead to different results due to the wide usage of the COVID-19 vaccines. Previous models learn from

[*] This work was done while the first author was doing internship at Baidu Inc.

the historical training data, which fail in modeling the time-variability of evolutional patterns after that.

Upon the above observations, we propose Complex Evolutional Network (CEN) to deal with the above two challenges. For length-diversity, CEN learns evolutional patterns from historical KG sequences of different lengths via an Relational Graph Neural Network (RGCN) based KG sequence encoder and a length-aware Convolutional Neural Network (CNN) based evolutional representation decoder. Besides, the model is trained via an easy-to-difficult curriculum learning strategy incrementally according to the length of KG sequences. For time-variability, we learn CEN under an online setting and combine CEN with a temporal regularization unit to alleviate the catastrophic forgetting problem (Mccloskey and Cohen, 1989).

In general, this paper makes the following contributions:

- We address, for the first time, the problems of length-diversity and time-variability of evolutional patterns for TKG reasoning.
- For length-diversity, we propose a length-aware CNN to learn evolutional patterns with different lengths in a curriculum learning manner. For time-variability, we propose to learn the model under an online setting to adapt to the changes of evolutional patterns.
- Experiments demonstrate that the proposed CEN model achieves better performance on TKG reasoning under both the traditional offline and the proposed online settings.

2 Related Work

The TKG reasoning task primarily has two settings, interpolation and extrapolation. This paper focus on the extrapolation setting. In what follows, we will introduce related work on both settings:

TKG Reasoning under the interpolation setting. This setting aims to complete the missing facts at past timestamps (Jiang et al., 2016; Leblay and Chekol, 2018; Dasgupta et al., 2018; Garcia-Duran et al., 2018; Goel et al., 2020; Wu et al., 2020). For example, TTransE (Leblay and Chekol, 2018) extends TransE (Bordes et al., 2013) by adding the temporal constraints; HyTE (Dasgupta et al., 2018) projects the entities and relations to time-aware hyperplanes to generate representations for different timestamps. Above all, they cannot obtain the representations of the unseen timestamps and are not suitable for the extrapolation setting.

TKG Reasoning under the extrapolation setting This setting aims to predict facts at future timestamps, which can be categorized into two groups: query-specific and entire graph based models. Query-specific models focus on modeling the query-specific history. For example, RE-NET (Jin et al., 2020) captures the evolutional patterns implied in the subgraph sequences of a fixed length specific to the query. CyGNet (Zhu et al., 2021) captures repetitive patterns by modeling repetitive facts. xERTE (Han et al., 2020a) learns to find the query-related subgraphs of a fixed hop number. CluSTeR (Li et al., 2021a) and TITer (Sun et al., 2021) both adopt reinforcement learning to discover evolutional patterns in query-related paths of a fixed length. Unlike the query-specific models, entire graph based models encode the latest historical KG sequence of a fixed-length. RE-GCN (Li et al., 2021b) captures the evolutional patterns into the representations of all the entities by modeling KG sequence of a fixed-length at lastest a few timestamps. Glean (Deng et al., 2020) introduces event descriptions to enrich the information of the entities.

3 Problem Formulation

A TKG $G = \{G_1, G_2, ..., G_t, ...\}$, where $G_t = (\mathcal{V}, \mathcal{R}, \mathcal{E}_t)$, is a directed multi-relational graph. $\mathcal{V}$ is the set of entities, $\mathcal{R}$ is the set of relations, and $\mathcal{E}_t$ is the set of facts at timestamp t. The TKG reasoning task aims to answer queries like $(s, r, ?, t_q)$ or $(?, r, o, t_q)$ with the historical KG sequence $\{G_1, G_2, ..., G_{t_q-1}\}$ given, where $s, o \in \mathcal{V}$, $r \in \mathcal{R}$ and t_q are the subject/object entity, the relation and the query timestamp, respectively. Following Jin et al. (2020), KGs from timestamps 1 to T_1, T_1 to T_2, T_2 to T_3 ($T_1 < T_2 < T_3$) are used as the training, validation and test sets, respectively. Under the traditional offline setting, models are trained only using the training set ($t_q \leq T_1$), while under the online setting, the model will be updated by KGs before t_q ($T_1 < t_q \leq T_3$) continually. Without loss of generality, we describe our model as predicting the missing object entity.

4 Methodology

We propose CEN to deal with the length-diversity and time-variability challenges of evolutional pat-

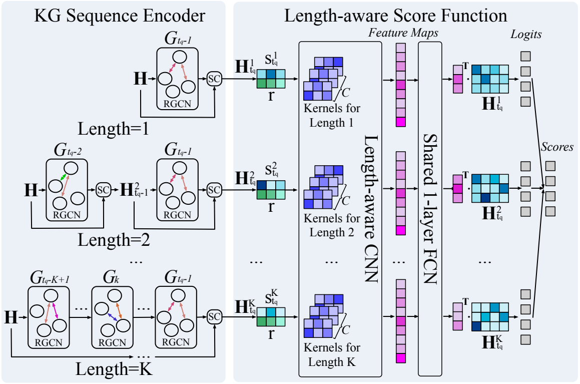


Figure 1: An diagram of the basic CEN model.

tern learning for TKG reasoning. Specifically, CEN consists of a basic model as well as a curriculum learning strategy for the former challenge and an online learning strategy for the latter challenge.

4.1 Basic CEN Model

As shown in Figure 1, the basic model of CEN contains a KG sequence encoder and an evolutional representation decoder. The KG sequence encoder encodes the latest historical KG sequences of different lengths to corresponding evolutional representations of entities. Then, the evolutional representation decoder calculates the scores of all entities for the query based on these representations.

KG Sequence Encoder. Its inputs include the lastest historical KG sequences of lengths from 1 to K, initial representations of entities $\mathbf{H} \in \mathbb{R}^{|\mathcal{V}|\times d}$ and relation representations $\mathbf{R} \in \mathbb{R}^{|\mathcal{R}|\times d}$, where d is the dimension of the representations. Take the KG sequence of length $k = 2$ for example, for each KG in the input sequence $\{G_{t_q-2}, G_{t_q-1}\}$, it iteratively calculates the evolutional representations of entities $\mathbf{H}^2_t$ at the corresponding timestamps $t \in \{t_q - 1, t_q\}$ as follows:

$$\hat{\mathbf{H}}^2_t = RGCN(\mathbf{H}^2_{t-1}, \mathbf{R}, G_{t-1}), \tag{1}$$

$$\mathbf{H}^2_t = SC(\hat{\mathbf{H}}^2_t, \mathbf{H}^2_{t-1}), \tag{2}$$

where $RGCN(\cdot)$ and SC denote the shared RGCN layer and the skip connection unit proposed in RE-GCN (Li et al., 2021b). For the initial timestamp $t_q - 1$, $\mathbf{H}^2_{t_q-2}$ is set to $\mathbf{H}$. $\mathbf{R}$ is shared across timestamps, which is different from RE-GCN. By reusing the encoder for KG sequences of different lengths, we obtain K entity evolution representations at the query timestamp: $\{\mathbf{H}^1_{t_q}, ..., \mathbf{H}^k_{t_q}, ..., \mathbf{H}^K_{t_q}\}$.

Evolutional Representation Decoder. Multiple evolutional representations contain evolutional patterns of multiple lengths. To distinguish the influences of the length-diverse evolutional patterns, we design a length-aware CNN, which uses K separate channels to model the above K evolutional representations. Specifically, for a query $(s, r, ?, t_q)$, the representations of s ($\mathbf{s}^1_{t_q}, ..., \mathbf{s}^k_{t_q}, ..., \mathbf{s}^K_{t_q}$) and r ($\mathbf{r}$) are looked up from multiple representations of entities $\{\mathbf{H}^1_{t_q}, ..., \mathbf{H}^k_{t_q}, ..., \mathbf{H}^K_{t_q}\}$ and the shared relation representations $\mathbf{R}$. For historical KG sequence of length k, k^{th} channel with C different kernels of size $2 \times M$ is used to decode the concatenation of $\mathbf{s}^k_{t_q}$ and $\mathbf{r}$. Specifically, the feature maps are calculated as below,

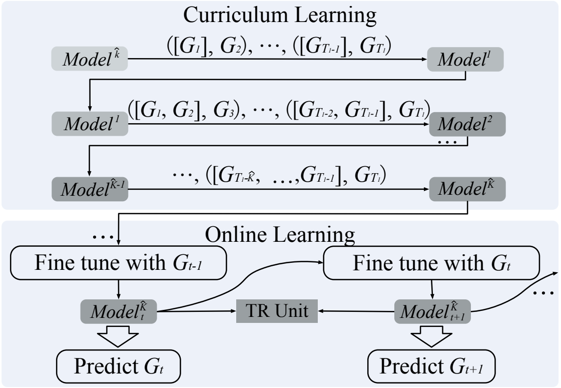


Figure 2: The learning procedure of the proposed model.

$$\mathbf{m}^k_c(s, r, t_q) = Conv_{2D}(\mathbf{w}^k_c, [\mathbf{s}^k_{t_q}; \mathbf{r}]), \tag{3}$$

where $Conv_{2D}$ denotes the 2D convolution operation, $\mathbf{w}^k_c$ $(0 \leq c < C)$ are the trainable parameters in c^{th} kernel of k^{th} channel and $\mathbf{m}^k_c(s, r, t_q) \in \mathbb{R}^{1\times d}$. After that, it concatenates the output vectors from C kernels yielding a vector: $\mathbf{m}^k(s, r, t_q) \in \mathbb{R}^{C\times d}$. For K channels, it outputs a list of vectors: $[\mathbf{m}^1(s, r, t_q)$,... , $\mathbf{m}^k(s, r, t_q)$,..., $\mathbf{m}^K(s, r, t_q)]$. Then, each vector is fed into a shared 1-layer Fully Connected Network (FCN) with $\mathbf{W}_3 \in \mathbb{R}^{Cd\times d}$ as its parameters and the final score of a candidate entity o is the sum of the logits from multiple evoltional representations: $\sum_{k=1}^{K} \mathbf{m}^k(s, r, t_q)\mathbf{W}_3\mathbf{o}^k$, where $\mathbf{o}^k$ is the evolutional representation of length k for o. Then we seen it as a multi-class learning problem and use the cross-entropy as its objective function.

4.2 Curriculum Learning for Length-diversity

Longer historical KG sequences contain more historical facts and longer evolutional patterns, which is more challenging to learn. Similar to human learning procedures, the models can benefit from an easy-to-difficult curriculum. Besides, how to

Datasets	ICEWS14	ICEWS18	WIKI
$\#\mathcal{E}$	6,869	23,033	12,554
$\#\mathcal{R}$	230	256	24
$\#Train$	74,845	373,018	539,286
$\#Valid$	8,514	45,995	67,538
$\#Test$	7,371	49,545	63,110
T_3	1 day	1 day	1 year

Table 1: Statistics of the datasets. $\#Train$, $\#Valid$, $\#Test$ are the numbers of facts in the training, validation and test sets.

choose the maximum length of evolutional patterns is vital to CEN. Thus, we design the curriculum learning strategy to learn the length-diverse evolutional patterns from short to long and adaptively select the optimal maximum length $\hat{K}$. As shown at the top of Figure 2, we start from the minimum length $\hat{k}$ ($\hat{k} = 1$ for example) and gradually move on to longer history in the training set. The model stops the curriculum and gets the optimal $\hat{K}$ when the MRR metric decreases or the length is up to maximum length K. Note that, curriculum learning is conducted under the traditional offline setting and $Model^{\hat{K}}$ is used as the pre-trained model for online learning.

4.3 Online Learning for Time-variability

To handle the time-variability of evolutional patterns, one simple and direct method is to update the model according to the newly occurred facts. Thus, as shown in the bottom of Figure 2, for timestamp $t+1$ ($T_1 < t+1 < T_3$), $Model_t^{\hat{K}}$ is fine-tuned to get $Model_{t+1}^{\hat{K}}$ by predicting the facts in the KG at the last timestamp G_t with historical KG sequences as inputs. Furthermore, to balance the knowledge of new evolutional patterns and the existing ones, we use a Temporal Regularization unit (TR unit) (Daruna et al., 2021; Wu et al., 2021). We apply an $L2$ regularization constraint between two temporally adjacent models to smooth the drastic change of the parameters.

4.4 Analysis on Computational Complexity

We analyze the computational complexity of CEN. We view the computational complexities of the RGCN unit and ConvTransE as constants. Then, the time complexity of the RGCN at a timestamp t is $O(|\mathcal{E}|)$, where $|\mathcal{E}|$ is the maximum number of facts at timestamps in history. As we unroll m ($m = \hat{K} - \hat{k}$) sequences, the time complexity of the KG sequence encoder is finally $O(m^2|\mathcal{E}|)$. Thus, the time complexity of CEN is $O(m^2|\mathcal{E}| + m)$.

5 Experiments

Experimental Setup. We adopt three widely-used datasets, ICEWS14 (Li et al., 2021b), ICEWS18 (Jin et al., 2020), and WIKI (Leblay and Chekol, 2018) to evaluate CEN. Dataset statistics are demonstrated in Table 1. Due to the space limitation, the CEN model is only compared with the latest models of TKG reasoning: CyGNet (Zhu et al., 2021), RE-NET (Jin et al., 2020), xERTE (Han et al., 2020a), TG-Tucker (Han et al., 2021), TG-DistMult (Han et al., 2021), TiTer (Sun et al., 2021) and RE-GCN (Li et al., 2021b). In the experiments, we adopt MRR (Mean Reciprocal Rank) and Hits@{1,3,10} as the metrics for TKG reasoning. We averaged the metrics over five runs. Note that, following Han et al. (2020b), we adopt an improved filtered setting where the timestamps of facts are considered, called time-aware filtered setting. Take a typical query $(s, r, ?, t_1)$ with answer o_1 in the test set for example, and assume there is another two facts (s, r, o_2, t_2) and (s, r, o_3, t_1). Under this time-aware filtered setting, only o_3 will be considered as a correct answer and thus removed from the ranking list of candidate answers.

Implementation Details. In the experiments, the optimal minimum lengths of evolutional patterns $\hat{k}$ for ICEWS14, ICEWS18, WIKI are 3, 3, 2, respectively. The maximum length K for all datasets is set to 10. For all datasets, the kernel width M is set to 3, and C is set to 50. For each fact (s, r, o, t) in the test set, we evaluate CEN on two queries $(s, r, ?, t)$ and $(?, r, o, t)$. The dimension d of relation representations and entity representations is set to 200 on all datasets. Adam (Kingma and Ba, 2014) is adopted for parameter learning with the learning rate of 0.001 on all datasets. The number of RGCN layers is set to 2 and the dropout rate for each layer to 0.2. For the online setting, we set the max epochs of the fine-tuning at each timestamp to 30. For predicting G_t, G_{t-2} is used as the validation set. We fine tune the pre-trained CEN from $T1 + 1$ to T_3 and report the results at the test timestamps (T_2 to T_3) in Table 3. The experiments are carried out on Tesla V100. Codes are avaliable at https://github.com/Lee-zix/CEN.

5.1 Experimental Results

Results under the Offline Setting. The results under the traditional offline setting are presented in Table 2. CEN consistently outperforms the

Model	ICEWS14				ICEWS18				WIKI			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
CyGNet	35.05	25.73	39.01	53.55	24.93	15.90	28.28	42.61	33.89	29.06	36.10	41.86
RE-NET	36.93	26.83	39.51	54.78	28.81	19.05	32.44	47.51	49.66	46.88	51.19	53.48
xERTE	40.02	32.06	44.63	56.17	29.31	21.03	33.51	46.48	71.14	68.05	76.11	79.01
TG-Tucker	-	-	-	-	28.68	19.35	32.17	47.04	50.43	48.52	51.47	53.58
TG-DistMult	-	-	-	-	26.75	17.92	30.08	44.09	51.15	49.66	52.16	53.35
TITer	40.97	**32.28**	45.45	57.10	29.98	**22.05**	33.46	44.83	75.50	72.96	77.49	79.02
RE-GCN	40.39	30.66	44.96	59.21	30.58	21.01	34.34	48.75	77.55	73.75	80.38	83.68
CEN	**42.20**	32.08	**47.46**	**61.31**	**31.50**	21.70	**35.44**	**50.59**	**78.93**	**75.05**	**81.90**	**84.90**

Table 2: Experimental results on TKG reasoning (in percentage) under the offline setting.

Model	ICEWS14				ICEWS18				WIKI			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
CEN(-TR)	39.28	30.05	43.58	57.01	31.11	21.41	35.09	50.27	**81.92**	**77.93**	**85.23**	**87.63**
CEN	**43.34**	**33.18**	**48.49**	**62.58**	**32.66**	**22.55**	**36.81**	**52.50**	79.67	75.63	83.00	85.58

Table 3: Experimental results on TKG reasoning (in percentage) under the online setting.

metrics	MRR	H@1	H@3	H@10
CEN	42.20	32.08	47.46	61.31
CEN(-CL)	41.50	31.53	46.50	60.81
CEN(-LA)	41.52	31.49	46.74	60.65

Table 4: Ablation Study of CEN on ICEWS14.

baselines on MRR, Hits@3, and Hits@10 on all datasets, which justifies the effectiveness of modeling the evolutional patterns of different lengths. On ICEWS datasets, CEN underperforms TITer on Hits@1 because TITer retrieves the answer through explicit paths, which usually gets high Hits@1. Whereas, CEN recalls more answer entities by aggregating the information from multiple evolutional patterns, which may be the reason for its high performance on Hits@3 and Hits@10.

Results under the Online Setting. Under the online setting, the model is updated via historical facts at the testset. Thus, it cannot be directly compared with the baselines designed for the offline setting. As shown in Table 3, on ICEWS datasets CEN outperforms CEN(-TR) (CEN without TR unit), which implies the effectiveness of TR unit to balance the knowledge of new evolutional patterns and the existing ones. On WIKI, CEN(-TR) gets better performance. It is because that the time interval between two adjacent timestamps in WIKI (one year) is much larger than ICEWS datasets (one day) and contains more time-variable evolutional patterns. TR unit limits the model to adapt to new knowledge and is not suitable for this dataset.

Ablation Study. To investigate the contributions of curriculum learning strategy and the length-aware CNN, we conduct ablation studies for CEN on the test set of ICEWS14 under the traditional offline setting, which are shown in Table 4. CEN(-CL) denotes CEN without the curriculum learning strategy. The underperformance of CEN(-CL) demonstrates the effectiveness of the curriculum learning strategy. CEN(-LA) denotes the model replacing the length-aware CNN with a traditional CNN. The underperformance of CEN(-LA) implies the effectiveness of the length-aware CNN.

6 Conclusions

In this paper, we proposed Complex Evolutional Network (CEN) for TKG reasoning, which deals with two challenges in modeling the complex evolutional patterns: length-diversity and time-variability. For length-diversity, CEN adopts a length-aware CNN to learn evolutional patterns of different lengths and is trained under a curriculum learning strategy. For time-variability, we explored a new online setting, where the model is expected to be updated to new evolutional patterns emerging over time. Experimental results demonstrate the superiority of the proposed model under both the offline and the online settings.

Acknowledgments

The work is supported by the National Natural Science Foundation of China under grants U1911401, 62002341 and 61772501, the GFKJ Innovation Program, Beijing Academy of Artificial Intelligence under grant BAAI2019ZD0306, and the Lenovo-CAS Joint Lab Youth Scientist Project.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. Icews coded event data. *Harvard Dataverse*, 12.

Angel Daruna, Mehul Gupta, Mohan Sridharan, and Sonia Chernova. 2021. Continual learning of knowledge graph embeddings. *IEEE Robotics and Automation Letters*, 6(2):1128–1135.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2001–2011.

Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2020. Dynamic knowledge graph based multi-event forecasting. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1585–1595.

Alberto Garcia-Duran, Sebastijan Dumančić, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4816–4821.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3988–3995.

Simon Gottschalk and Elena Demidova. 2018. Eventkg: A multilingual event-centric temporal knowledge graph. In *European Semantic Web Conference*, pages 272–287. Springer.

Simon Gottschalk and Elena Demidova. 2019. Eventkg–the hub of event knowledge on the web–and biographical timeline generation. *Semantic Web*, (Preprint):1–32.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2020a. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *International Conference on Learning Representations*.

Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. 2021. Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8352–8364, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhen Han, Yunpu Ma, Yuyi Wang, Stephan Günnemann, and Volker Tresp. 2020b. Graph hawkes neural network for forecasting on temporal knowledge graphs. *8th Automated Knowledge Base Construction (AKBC)*.

Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Sujian Li, Baobao Chang, and Zhifang Sui. 2016. Encoding temporal information for time-aware link prediction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2350–2354.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020. Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6669–6683.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. In *Companion Proceedings of the The Web Conference 2018*, pages 1771–1776. International World Wide Web Conferences Steering Committee.

Zixuan Li, Xiaolong Jin, Saiping Guan, Wei Li, Jiafeng Guo, Yuanzhuo Wang, and Xueqi Cheng. 2021a. Search from history and reason for future: Two-stage reasoning on temporal knowledge graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4732–4743.

Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. 2021b. Temporal knowledge graph reasoning based on evolutional representation learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 408–417.

M. Mccloskey and N. J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165.

Haohai Sun, Jialun Zhong, Yunpu Ma, Zhen Han, and Kun He. 2021. Timetraveler: Reinforcement learning for temporal knowledge graph forecasting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8306–8319.

Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L Hamilton. 2020. Temp: Temporal message passing for temporal knowledge graph completion. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5730–5746.

Jiapeng Wu, Yishi Xu, Yingxue Zhang, Chen Ma, Mark J Coates, and Jackie Chi Cheung. 2021. Tie: A framework for embedding-based incremental temporal knowledge graph completion.

Liang Zhao. 2020. Event prediction in big data era: A systematic survey. *arXiv preprint arXiv:2007.09815*.

Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. 2021. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4732–4740.

Mismatch between Multi-turn Dialogue and its Evaluation Metric in Dialogue State Tracking

Takyoung Kim[1], **Hoonsang Yoon**[1], **Yukyung Lee**[1], **Pilsung Kang**[1], **Misuk Kim**[2]
[1]Korea University, Seoul, Republic of Korea
[2]Sejong University, Seoul, Republic of Korea
[1]{takyoung_kim, hoonsang_yoon, yukyung_lee, pilsung_kang}@korea.ac.kr
[2]misuk.kim@sejong.ac.kr

Abstract

Dialogue state tracking (DST) aims to extract essential information from multi-turn dialogue situations and take appropriate actions. A belief state, one of the core pieces of information, refers to the subject and its specific content, and appears in the form of `domain-slot-value`. The trained model predicts "accumulated" belief states in every turn, and joint goal accuracy and slot accuracy are mainly used to evaluate the prediction; however, we specify that the current evaluation metrics have a critical limitation when evaluating belief states accumulated as the dialogue proceeds, especially in the most used MultiWOZ dataset. Additionally, we propose **relative slot accuracy** to complement existing metrics. Relative slot accuracy does not depend on the number of predefined slots, and allows intuitive evaluation by assigning relative scores according to the turn of each dialogue. This study also encourages not solely the reporting of joint goal accuracy, but also various complementary metrics in DST tasks for the sake of a realistic evaluation.

1 Introduction

The dialogue state tracking (DST) module structures the belief state that appears during the conversation in the form of `domain-slot-value`, to provide an appropriate response to the user. Recently, multi-turn DST datasets have been constructed using the Wizard-of-Oz method to reflect more realistic dialogue situations (Wen et al., 2017; Mrkšić et al., 2017; Budzianowski et al., 2018). The characteristic of these datasets is that belief states are "accumulated" and recorded every turn. That is, the belief states of the previous turns are included in the current turn. It confirms whether the DST model tracks essential information that has appeared up to the present point.

Joint goal accuracy and slot accuracy are utilized in most cases to evaluate the prediction of accumulated belief states. Joint goal accuracy strictly determines whether every predicted state is identical to the gold state, whereas slot accuracy measures the ratio of correct predictions. However, we determined that these two metrics solely focus on "penalizing states that fail to predict," not considering "reward for well-predicted states." Accordingly, as also pointed out in Rastogi et al. (2020a), joint goal accuracy underestimates the model prediction because of its error accumulation attribute, while slot accuracy overestimates it because of its dependency on predefined slots.

However, there is a lack of discussion on the metric for evaluating the most used MultiWOZ dataset, despite a recently published dataset (Rastogi et al., 2020b) proposing some metrics. To address the above challenge, we propose reporting the **relative slot accuracy** along with the existing metrics in MultiWOZ dataset. While slot accuracy has the challenge of overestimation by always considering all predefined slots in every turn, relative slot accuracy does not depend on predefined slots, and calculates a score that is affected solely by slots that appear in the current dialogue. Therefore, relative slot accuracy enables a realistic evaluation by rewarding the model's correct predictions, a complementary approach that joint goal and slot accuracies cannot fully cover. It is expected that the proposed metric can be adopted to evaluate model performance more intuitively.

2 Current Evaluation Metrics

2.1 Joint Goal Accuracy

Joint goal accuracy, developed from Henderson et al. (2014b) and Zhong et al. (2018), can be said to be an ideal metric, in that it verifies that the predicted belief states perfectly match the gold label. Equation 1 expresses how to calculate the joint goal accuracy, depending on whether the slot values match each turn.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 297 - 309
May 22-27, 2022

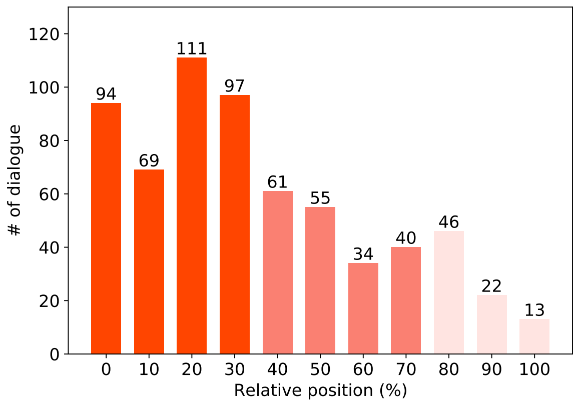


Figure 1: The relative position where joint goal accuracy of the turn is measured to be zero for the first time among the dialogues where joint goal accuracy of the last turn is zero. (642 of 999 MultiWOZ 2.1 test set with SOM-DST).

$$JGA = \begin{cases} 1 & \text{if predicted state} = \text{gold state} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

However, the joint goal accuracy underestimates the accumulated states because it scores the performances of later turn to zero if the model mispredicts even once in a particular turn, regardless of the model prediction quality at later turns. As illustrated in Figure 1, we measured the relative position of the turn causing this phenomenon for the dialogue. We used MultiWOZ 2.1 (Eric et al., 2019), and analyzed 642 samples from a total of 999 test sets in which the joint goal accuracy of the last turn is zero. The DST model selected for primary verification is the SOM-DST (Kim et al., 2020), which is one of the latest DST models. Accordingly, the relative position where joint goal accuracy first became zero was mainly at the beginning of the dialogue[1]. This means that the joint goal accuracy after the beginning of the dialogue is unconditionally measured as zero because of the initial misprediction, although the model may correctly predict new belief states at later turns. Failure to measure the performance of the latter part means that it cannot consider various dialogue situations provided in the dataset, which is a critical issue in building a realistic DST model.

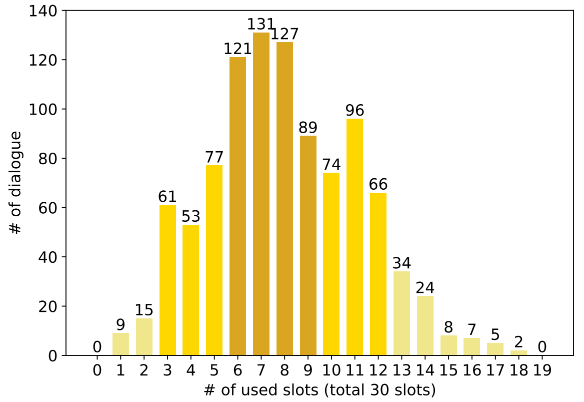


Figure 2: The number of predefined gold slots used in each dialogue (999 MultiWOZ 2.1 test set).

2.2 Slot Accuracy

Slot accuracy can compensate for situations where joint goal accuracy does not fully evaluate the dialogue situation. Equation 2 expresses how to calculate the slot accuracy. T indicates the total number of predefined slots for all the domains. M denotes the number of missed slots that the model does not accurately predict among the slots included in the gold state, and W denotes the number of wrongly predicted slots among the slots that do not exist in the gold state.

$$SA = \frac{T - M - W}{T} \tag{2}$$

Figure 2 illustrates the total number of annotated slots in MultiWOZ 2.1 to figure out the limitation of slot accuracy. Each value of x-axis in Figure 2 indicates the "maximum" number of slots that appear in a single dialogue, and we confirmed that approximately 85% of the test set utilized solely less than 12 of the 30 predefined slots in the experiment. Because the number of belief states appearing in the early and middle turns of the dialogue are smaller, and even fewer states make false predictions, calculating slot accuracy using Equation 2 reduces the influence of M and W, and the final score is dominated by the total slot number T. Accordingly, several previous studies still report the model performance using solely joint goal accuracy because slot accuracy excessively depends on the number of predefined slots, making the performance deviation among models trivial (refer to Table A5).

Furthermore, according to Table A6, we determined that slot accuracy tends to be too high. The slot accuracies of turns 0 and 1 show approximately 96% accuracy, despite the model not cor-

[1] 59 samples of the 642 samples have a joint goal accuracy of 1 in the middle, owing to a coincidental situation or differences in the analysis of annotation. Table A1 and Table A2 show the dialogue situation in detail, and Table A3 and Table A4 show the belief states accordingly. Refer to Appendix A.

Type	Model	Joint Goal Acc.	Slot Acc.	F1 Score	Relative Slot Acc.
Open vocabulary	Transformer-DST (2021)	0.5446	0.9748	0.9229	0.8759
	TripPy (2020)	**0.6131**	0.9707	0.8573	0.8432
	SOM-DST (2020)	0.5242	0.9735	0.9179	0.8695
	Simple-TOD (2020)	0.5605	**0.9761**	**0.9276**	**0.8797**
	SAVN (2020)	0.5357	0.9749	0.9246	0.8769
	TRADE (2019)	0.4939	0.9700	0.9033	0.8520
	COMER (2019)	0.4879	0.9652	0.8800	0.8250
Ontology based	DST-STAR (2021)	0.5483	0.9754	0.9253	0.8780
	L4P4K2-DSGraph (2021)	0.5178	0.9690	0.9189	0.8570
	SUMBT (2019)	0.4699	0.9666	0.8934	0.8380

Table 1: Model performance of MultiWOZ 2.1 with various evaluation metrics. All reported performances are our re-implementation.

rectly predicting states at all. It becomes difficult to compare various models in detail, if each model shows a high performance, even though nothing is adequately predicted. In addition, as the turn progresses, there are no rewards for a situation in which the model tracks the belief state without any challenges. The case correctly predicting two out of three in turn 4, and the case correctly predicting three out of four in turn 5 exhibit the same slot accuracy. Therefore, the slot accuracy measured according to Equation 2 differs from our intuition.

2.3 Other Metric

Recently, Rastogi et al. (2020b) proposed a metric called average goal accuracy. The main difference between the average goal accuracy and the proposed relative slot accuracy is that the average goal accuracy only considers the slots with non-empty values in the gold states of each turn, whereas the proposed relative slot accuracy considers those in both gold and predicted states. Since average goal accuracy ignores the predicted states, it cannot properly distinguish a better model from a worse model in some specific situations. We will discuss it in more detail in Section 4.1.

3 Relative Slot Accuracy

As can be observed in Equation 2, slot accuracy has the characteristic that the larger the number of predefined slots (T), the smaller the deviation between the prediction results. The deviation among DST models will be even more minor when constructing datasets with various dialogue situations, because the number of predefined slots will continually increase. It is not presumed to be an appropriate metric in terms of scalability.

Therefore, we propose relative slot accuracy, that is not affected by predefined slots, and is evaluated with adequate rewards and penalties that fit human intuition in every turn. Equation 3 expresses how to calculate the relative slot accuracy, and T^* denotes the number of unique slots appearing in the predicted and gold states in a particular turn.

$$RSA = \frac{T^* - M - W}{T^*}, \text{ where } 0 \text{ if } T^* = 0 \quad (3)$$

Relative slot accuracy rewards well-predicted belief states by measuring the scores in accumulating turns. Further discussions on the relative score will be discussed in Section 4.1.

4 Experiments

We measured MultiWOZ 2.1, an improved version of MultiWOZ 2.0 (Budzianowski et al., 2018), which has been adopted in several studies, according to Table A5. Five domains (i.e., *hotel, train, restaurant, attraction,* and *taxi*) are adopted in the experiment, following Wu et al. (2019), and there are a total of 30 `domain-slot` pairs. We selected the DST models in Table A5 that perform the MultiWOZ experiment with the original authors' reproducible code[2]. Additionally, we reported the F1 score, which can be calculated using the current predicted and gold states.

[2] Implementation codes for Simple-TOD and TripPy are from https://github.com/salesforce/coco-dst.

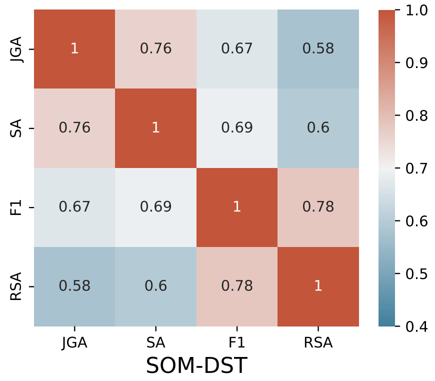


Figure 3: Correlation matrix of evaluation performance of total 7,368 turns in 999 MultiWOZ 2.1 test set using SOM-DST. Results for other models are included in Figure A1.

4.1 Results and Discussion

Table 1 presents the overall results. Regarding slot accuracy, the difference between the largest and smallest values is solely 1.09%. It can be one of the reasons that several researchers do not report it. Meanwhile, relative slot accuracy can explicitly highlight the deviation among models by showing a 5.47% difference between the largest and smallest values. Furthermore, the correlation with joint goal accuracy, a mainly adopted metric, and relative slot accuracy with respect to each turn is lower than the correlation with joint goal accuracy and slot accuracy, as illustrated in Figure 3. Specifically, it can be compared with a different perspective when using the proposed reward-considering evaluation metric.

Domain-specific Evaluation We reported the joint goal, slot, and relative slot accuracies per domain utilizing the SOM-DST model in Table 2. Relative slot accuracy derives a specific score in the turn configuration and prediction ratio of each domain by excluding slots that do not appear in the conversation. For example, the *taxi* domain shows a low score, meaning that it has relatively several cases of incorrect predictions, compared to the number of times slots belonging to the *taxi* domain appear. Because slot accuracy cannot distinguish the above trend, the score of the *hotel* domain is lower than that of the *taxi* domain. In summary, relative slot accuracy enables relative comparison according to the distribution of the domain in a dialogue.

Domain	Joint Goal Acc.	Slot Acc.	Relative Slot Acc.
hotel	0.4923	0.9731	0.8493
train	0.7162	0.9874	0.9176
restaurant	0.6589	0.9858	0.8977
attraction	0.6811	0.9878	0.8421
taxi	0.5701	0.9798	0.7828

Table 2: Per-domain performance of SOM-DST prediction.

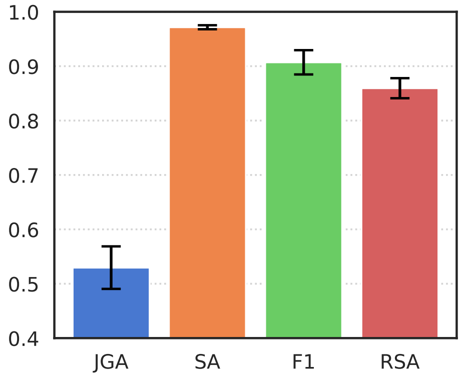


Figure 4: The mean and standard deviation of model performance reported in Table 1.

Dependency on Predefined Slots As discussed in Section 2.2, slot accuracy requiring total predefined slots is not a scalable method for evaluating the current dialogue dataset that contains a few domains in each dialogue. For example, when evaluating a dialogue sample that solely deals with the *restaurant* domain, even domains that never appear at all (i.e., *hotel, train, attraction,* and *taxi*) are involved in measuring performance, making deviations among different models trivial. However, relative slot accuracy can evaluate the model's predictive score without being affected by slots never seen in the current dialogue, which is a more realistic way, considering that each dialogue contains its own turn and slot composition. Figure 4 illustrates the mean and standard deviations of the model performance in Table 1. As can be observed from the results, the relative slot accuracy has a higher deviation than the slot accuracy, enabling a detailed comparison among the methodologies.

Reward on Relative Dialogue Turn Relative slot accuracy is able to reward the model's correct prediction by measuring the accuracy on a relative basis for each turn. Table A6 compares the slot and relative slot accuracies. The relative slot accuracy from turns 0 – 3 is measured as 0 because it cal-

Type	Belief State	Joint Goal Acc.	Average Goal Acc.	Relative Slot Acc.
Gold State	restaurant-area-centre restaurant-food-indian restaurant-people-2	-	-	-
Prediction of Model A	restaurant-area-centre restaurant-food-chinese attraction-area-centre	0	0.3333	0.2500
Prediction of Model B	restaurant-area-centre restaurant-food-chinese restaurant-name-nusha attraction-area-centre attraction-pricerange-cheap	0	0.3333	0.1667

Table 3: A situation that average goal accuracy cannot distinguish between two models. States with blue denote correct prediction, and as defined in Section 2.2, states with orange and pink denote respective M and W.

culates the score based on the unique state of the current turn according to Equation 3. In addition, regarding slot accuracy in turns 4, 5, and 6, there is no score improvement for the additional well-predicted state by the model, whereas the score increases when the newly added state is matched in the case of relative slot accuracy. Therefore, relative slot accuracy can provide an intuitive evaluation reflecting the current belief state recording method, in which the number of slots accumulates incrementally as the conversation progresses.

Comparison to Average Goal Accuracy Relative slot accuracy can compare DST model performances more properly than average goal accuracy, as mentioned in Section 2.3. Table 3 describes how these two metrics result in different values for the same model predictions. In this example, average goal accuracy cannot consider additional belief states incorrectly predicted by `Model B`, resulting in the same score between the two models. In contrast, relative slot accuracy can give a penalty proportional to the number of wrong predictions because it includes both gold and predicted states when calculating the score. Consequently, relative slot accuracy has a more elaborated discriminative power than the average goal accuracy.

5 Conclusion

This paper points out the challenge that the existing joint goal and slot accuracies cannot fully evaluate the accumulating belief state of each turn in the MultiWOZ dataset. Accordingly, the relative slot accuracy is proposed. This metric is not affected by unseen slots in the current dialogue situation, and compensates for the model's correct prediction. When the DST task is scaled up to deal with more diverse conversational situations, a realistic model evaluation will be possible using relative slot accuracy. Moreover, we suggest reporting various evaluation metrics to complement the limitations of each metric in future studies, not solely reporting the joint goal accuracy.

Acknowledgement

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2021-0-00034, Clustering technologies of fragmented data for time-based data analysis) and Ministry of Culture, Sports and Tourism and Korea Creative Content Agency (Project Number: R2020040126-0001)

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Guan-Lin Chao and Ian R. Lane. 2019. BERT-DST: scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 1468–1472. ISCA.

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. 2020. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *AAAI*.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tür. 2019. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *CoRR*, abs/1907.01669.

Yue Feng, Yang Wang, and Hang Li. 2021. A sequence-to-sequence approach to dialogue state tracking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1714–1725, Online. Association for Computational Linguistics.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tur. 2019. Dialog state tracking: A neural reading comprehension approach. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 264–273, Stockholm, Sweden. Association for Computational Linguistics.

Rahul Goel, Shachi Paul, and Dilek Hakkani-Tür. 2019. Hyst: A hybrid approach for flexible and accurate dialogue state tracking. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 1458–1462. ISCA.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. TripPy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 35–44, 1st virtual meeting. Association for Computational Linguistics.

Matthew Henderson, Blaise Thomson, and Jason D Williams. 2014a. The second dialog state tracking challenge. In *Proceedings of the 15th annual meeting of the special interest group on discourse and dialogue (SIGDIAL)*, pages 263–272.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014b. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 292–299, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A Simple Language Model for Task-Oriented Dialogue. In *Advances in Neural Information Processing Systems*, volume 33, pages 20179–20191. Curran Associates, Inc.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 567–582, Online. Association for Computational Linguistics.

Hung Le, Richard Socher, and Steven C.H. Hoi. 2020. Non-autoregressive dialog state tracking. In *International Conference on Learning Representations*.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. SUMBT: Slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5478–5483, Florence, Italy. Association for Computational Linguistics.

Weizhe Lin, Bo-Hsiang Tseng, and Bill Byrne. 2021. Knowledge-aware graph-enhanced gpt-2 for dialogue state tracking.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, Vancouver, Canada. Association for Computational Linguistics.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020a. Schema-guided dialogue state tracking task at dstc8.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020b. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Liliang Ren, Jianmo Ni, and Julian McAuley. 2019. Scalable and accurate dialogue state tracking via hierarchical sequence generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1876–1885, Hong Kong, China. Association for Computational Linguistics.

Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. 2018. Building a conversational agent overnight with dialogue self-play.

Yexiang Wang, Yi Guo, and Siqi Zhu. 2020. Slot attention with value normalization for multi-domain dialogue state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3019–3028, Online. Association for Computational Linguistics.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational*

Linguistics: Volume 1, Long Papers, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819, Florence, Italy. Association for Computational Linguistics.

Peng Wu, Bowei Zou, Ridong Jiang, and AiTi Aw. 2020. GCDST: A graph-based and copy-augmented multi-domain dialogue state tracking. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1063–1073, Online. Association for Computational Linguistics.

Fanghua Ye, Jarana Manotumruksa, Qiang Zhang, Shenghui Li, and Emine Yilmaz. 2021. Slot self-attentive dialogue state tracking. In *Proceedings of the Web Conference 2021*, WWW '21, page 1598–1608, New York, NY, USA. Association for Computing Machinery.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2 : A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117, Online. Association for Computational Linguistics.

Yan Zeng and Jian-Yun Nie. 2021. Jointly optimizing state operation prediction and value generation for dialogue state tracking.

Jianguo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wang, Philip Yu, Richard Socher, and Caiming Xiong. 2020. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 154–167, Barcelona, Spain (Online). Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1458–1467, Melbourne, Australia. Association for Computational Linguistics.

Li Zhou and Kevin Small. 2019. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *CoRR*, abs/1911.06192.

Su Zhu, Jieyu Li, Lu Chen, and Kai Yu. 2020. Efficient context and schema fusion networks for multi-domain dialogue state tracking. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 766–781, Online. Association for Computational Linguistics.

A Complementary discussions of joint goal accuracy

Our findings show that if the model makes an incorrect prediction, the error accumulates until the end of the dialogue, and the joint goal accuracy remains at zero. In this section, we discuss a few cases of 59 dialogues that do not show the trend among 642 dialogues selected in Section 2.1; however, it is important to note that these few cases have negligible effect on the trend in Figure 1, solely changing the position where the joint goal accuracy first becomes zero.

We sampled dialogues of the MultiWOZ 2.1 test set in Table A1 and Table A2, and marked values appearing in the dialogue in bold. Table A3 and Table A4 indicate the corresponding belief states of each dialogue. In the first dialogue presented in Table A1, the joint goal accuracy is measured as 1 at turn 2. In this case, the model incorrectly predicted the `restaurant-pricerange` slot at turns 0 and 1, and then the utterance about the slot appeared by chance. In a general case, the wrong prediction of the `restaurant-pricerange` slot at turn 0 will accumulate to the last turn. However, in this case, another incorrect prediction at turn 3 will cause error accumulation in this dialogue.

The second dialogue presented in Table A2, reports the incorrect prediction according to the interpretation of annotations at turn 4. In other words, because the dialogue about the `hotel-internet` slot appears over turns 4 and 5, it is solely an error depending on the prediction timing of the model. Because the correct belief state was predicted right from turn 5, it cannot be said to be an error accumulation phenomenon; however, the model did not predict the `hotel-pricerange` slot at turn 6, which is the last turn in this case.

In conclusion, it can be determined that the model does not seem to accumulate erroneous predictions because of an accidental situation or interpretation of annotations, but this does not negate the error accumulation phenomenon. Furthermore, the fact that the starting point of making the joint goal accuracy of subsequent turns to 0 mainly occurs at the beginning of the dialogue does not change.

Turn	Dialogue History
0	*System*: “ ” *User*: “can you help me find a nice restaurant ?”
1	*System*: “sure ! what kind of food do you like ?” *User*: “i was thinking some **indian** food would be great .”
2	*System*: “i have 22 indian restaurant -s do you have a preference for area of town ?” *User*: “no , i **do not care** where it is . i am thinking i would like an **expensive** restaurant .”
3	*System*: “would you like to try **curry garden** ?” *User*: “that is fine book me a table for **6** on **sat** at **17:30** . i also need a **train for the same** and should **leave leicester** for **cambridge**”
4	*System*: “alright , i have made your requested booking at curry garden , and the reference number is hk9ycl6z . as for the train , what time would you like to depart ?” *User*: “it does not matter as long as i am there by **13:45** leaving leicester going to cambridge , i’ll need the reference number too please”
5	*System*: “i have found tr6210 leaving leicester at 11:09 on saturday and arriving in cambridge at 12:54 . i have booked 6 seats . reference number is lr5i1rzv . anything else i can do ?” *User*: “that will be all for today . thanks so much for all your help . goodbye”

Table A1: Sample dialogue of MultiWOZ 2.1 test set (PMUL4234.json).

Turn	Dialogue History
0	*System*: “ ” *User*: “i would like help finding a train headed to **cambridge** .”
1	*System*: “i will be happy to help you find a train . can you tell me where you will be departing from ?” *User*: “departing from **london kings cross** on **tuesday** .”
2	*System*: “when would you like to leave or arrive by ?” *User*: “i need to **arrive by 18,30** .”
3	*System*: “take train tr1434 , which will arrive at 18:08 . shall i book you for that train ?” *User*: “can i get the price for a ticket , first ?”
4	*System*: “sure ! the ticket is 23.60 pounds .” *User*: “thanks ! i am also looking for a hotel called **archway house** . can you tell me if they have **free wifi** ?”
5	*System*: “they do . would you like to book a room ?” *User*: “i would first like to know what their price range and hotel type are , thank you .”
6	*System*: “archway house is a **moderate** -ly priced guesthouse . would you like their address or perhaps to book a room there ?” *User*: “thank you , but no . you’ve already helped me with everything i needed today .”

Table A2: Sample dialogue of MultiWOZ 2.1 test set (MUL2270.json).

Turn	Predicted State	Gold State	Joint Goal Acc.
0	restaurant-pricerange-expensive	-	0
1	restaurant-pricerange-expensive restaurant-food-indian	restaurant-food-indian	0
2	restaurant-pricerange-expensive restaurant-food-indian restaurant-area-dontcare	restaurant-pricerange-expensive restaurant-food-indian restaurant-area-dontcare	1
3	restaurant-pricerange-expensive restaurant-food-indian restaurant-area-dontcare restaurant-book day-sunday restaurant-book people-6 restaurant-book time-17:30 restaurant-name-curry garden train-destination-cambridge train-day-tuesday train-departure-leicester	restaurant-pricerange-expensive restaurant-food-indian restaurant-area-dontcare restaurant-book day-saturday restaurant-book people-6 restaurant-book time-17:30 restaurant-name-curry garden train-destination-cambridge train-day-saturday train-departure-leicester train-book people-6	0
4	restaurant-pricerange-expensive restaurant-food-indian restaurant-area-dontcare restaurant-book day-sunday restaurant-book people-6 restaurant-book time-17:30 restaurant-name-curry garden train-destination-cambridge train-day-tuesday train-departure-leicester train-arriveby-13:45 train-leaveat-dontcare	restaurant-pricerange-expensive restaurant-food-indian restaurant-area-dontcare restaurant-book day-saturday restaurant-book people-6 restaurant-book time-17:30 restaurant-name-curry garden train-destination-cambridge train-day-saturday train-departure-leicester train-arriveby-13:45 train-book people-6	0
5	restaurant-pricerange-expensive restaurant-food-indian restaurant-area-dontcare restaurant-book day-sunday restaurant-book people-6 restaurant-book time-17:30 restaurant-name-curry garden train-destination-cambridge train-day-tuesday train-departure-leicester train-arriveby-13:45 train-leaveat-dontcare	restaurant-pricerange-expensive restaurant-food-indian restaurant-area-dontcare restaurant-book day-saturday restaurant-book people-6 restaurant-book time-17:30 restaurant-name-curry garden train-destination-cambridge train-day-saturday train-departure-leicester train-arriveby-13:45 train-book people-6	0

Table A3: SOM-DST prediction of MultiWOZ 2.1 test sample (PMUL4234.json).

Turn	Predicted State	Gold State	Joint Goal Acc.
0	train-destination-cambridge	train-destination-cambridge	1
1	train-destination-cambridge train-day-tuesday train-departure-london kings cross	train-destination-cambridge train-day-tuesday train-departure-london kings cross	1
2	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30	1
3	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30	1
4	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30 hotel-name-archway house	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30 hotel-name-archway house hotel-internet-yes	0
5	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30 hotel-name-archway house hotel-internet-yes	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30 hotel-name-archway house hotel-internet-yes	1
6	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30 hotel-name-archway house hotel-internet-yes	train-destination-cambridge train-day-tuesday train-departure-london kings cross train-arriveby-18:30 hotel-name-archway house hotel-internet-yes hotel-pricerange-moderate	0

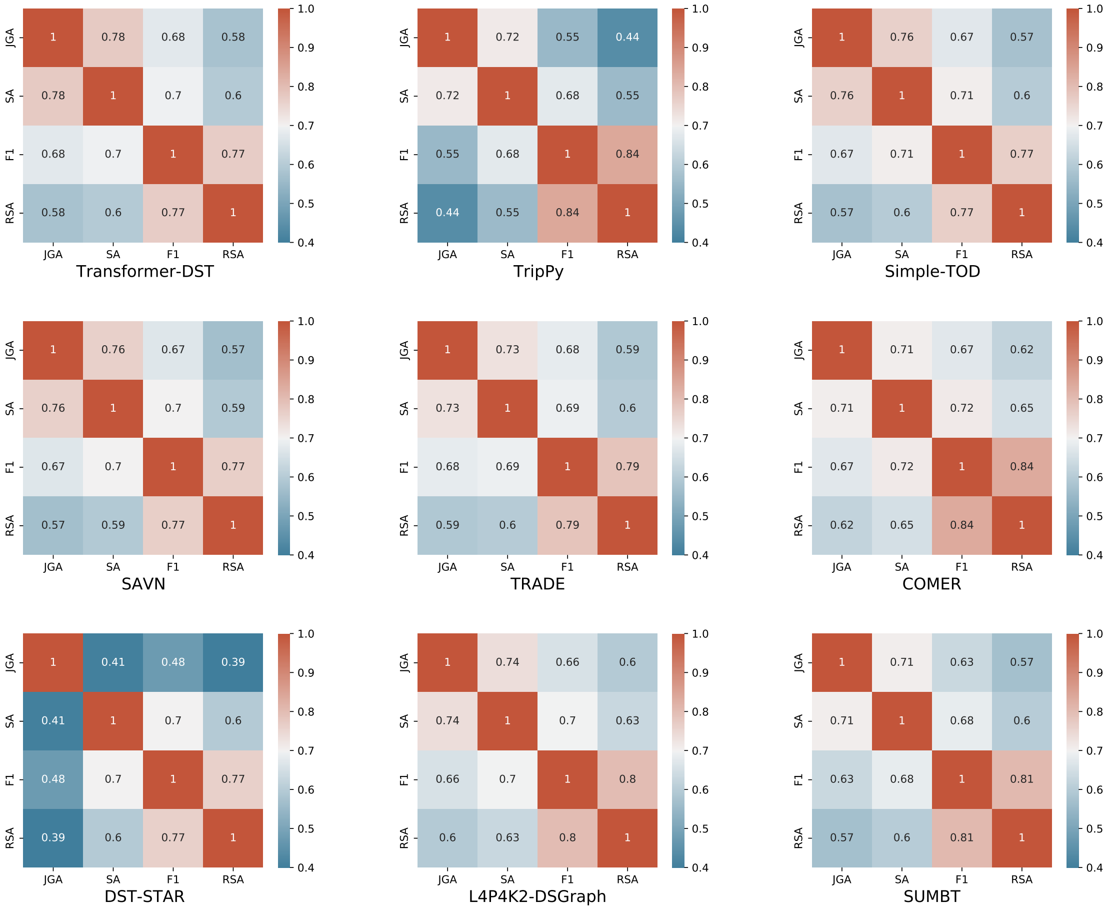
Table A4: SOM-DST prediction of MultiWOZ 2.1 test sample (MUL2270.json).

Method	**Metric**	**Dataset**
DST-STAR (Ye et al., 2021)	JGA	MultiWOZ 2.0 (Budzianowski et al., 2018), MultiWOZ 2.1 (Eric et al., 2019)
Seq2Seq-DU (Feng et al., 2021)	JGA	SGD (Rastogi et al., 2020b), MultiWOZ 2.1, MultiWOZ 2.2 (Zang et al., 2020)
L4P4K2-DSGraph (Lin et al., 2021)	JGA, SA	MultiWOZ 2.0
Transformer-DST (Zeng and Nie, 2021)	JGA	MultiWOZ 2.0, MultiWOZ 2.1
NA-DST (Le et al., 2020)	JGA, SA	MultiWOZ 2.0, MultiWOZ 2.1
TripPy (Heck et al., 2020)	JGA	WOZ 2.0 (Wen et al., 2017), MultiWOZ 2.1, Sim-M, Sim-R (Shah et al., 2018)
SOM-DST (Kim et al., 2020)	JGA, SA	MultiWOZ 2.0, MultiWOZ 2.1
Simple-TOD (Hosseini-Asl et al., 2020)	JGA	MultiWOZ 2.0, MultiWOZ 2.1
GCDST (Wu et al., 2020)	JGA	MultiWOZ 2.0, MultiWOZ 2.1
CSFN-DST (Zhu et al., 2020)	JGA	MultiWOZ 2.0, MultiWOZ 2.1
SAVN (Wang et al., 2020)	JGA, SA	MultiWOZ 2.0, MultiWOZ 2.1
SST (Chen et al., 2020)	JGA, SA	MultiWOZ 2.0, MultiWOZ 2.1
DS-DST (Zhang et al., 2020)	JGA	MultiWOZ 2.0, MultiWOZ 2.1
DSTQA (Zhou and Small, 2019)	JGA, SA	WOZ 2.0, MultiWOZ 2.0, MultiWOZ 2.1
SUMBT (Lee et al., 2019)	JGA	WOZ 2.0, MultiWOZ 2.0
DST-Reader (Gao et al., 2019)	JGA	MultiWOZ 2.0
BERT-DST (Chao and Lane, 2019)	JGA	WOZ 2.0, Sim-M, Sim-R DSTC2 (Henderson et al., 2014a)
TRADE (Wu et al., 2019)	JGA, SA	MultiWOZ 2.0
HyST (Goel et al., 2019)	JGA	MultiWOZ 2.0
COMER (Ren et al., 2019)	JGA	WOZ 2.0, MultiWOZ 2.0

Table A5: Evaluation metrics used for performance comparison among the methodologies. We focused on metrics evaluating the belief state of each turn. For convenience, the name of each metric is abbreviated. JGA: Joint Goal Accuracy, SA: Slot Accuracy.

Turn	Predicted State	Gold State	Slot Acc.	Relative Slot Acc.
0	restaurant-name-nusha	-	0.9667	0
1	restaurant-name-nusha	-	0.9667	0
2	restaurant-name-nusha	attraction-name-nusha	0.9333	0
3	restaurant-name-nusha	attraction-name-nusha	0.9333	0
4	restaurant-area-centre restaurant-food-indian	attraction-name-nusha restaurant-area-centre restaurant-food-indian	0.9667	0.6667
5	restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	attraction-name-nusha restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	0.9667	0.7500
6	restaurant-name-saffron brasserie restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	attraction-name-nusha restaurant-name-saffron brasserie restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	0.9667	0.8000
7	restaurant-name-saffron brasserie restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	attraction-name-nusha restaurant-name-saffron brasserie restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	0.9667	0.8000
8	restaurant-name-saffron brasserie restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	attraction-name-nusha restaurant-name-saffron brasserie restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	0.9667	0.8000
9	restaurant-name-saffron brasserie restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	attraction-name-nusha restaurant-name-saffron brasserie restaurant-area-centre restaurant-food-indian restaurant-pricerange-expensive	0.9667	0.8000

Table A6: SOM-DST prediction of MultiWOZ 2.1 test sample (PMUL4648.json). The joint goal accuracy of every turn is 0 because of belief states with red color. When calculating score, the number of total slots is set to 30, which is of *hotel, train, restaurant, attraction,* and *taxi* domains in MultiWOZ 2.1. Relative slot accuracy can be calculated just using slot-values appearing in the dialogue, not being affected by unused information.



Transformer-DST

TripPy

Simple-TOD

SAVN

TRADE

COMER

DST-STAR

L4P4K2-DSGraph

SUMBT

Figure A1: Correlation matrices of evaluation performance using various DST models.

LM-BFF-MS: Improving Few-Shot Fine-tuning of Language Models based on Multiple Soft Demonstration Memory

Eunhwan Park[†], **Donghyeon Jeon**[‡], **Seonhoon Kim**[‡],
Inho Kang[‡], **Seung-Hoon Na**[†*]
[†]Jeonbuk National University, [‡]NAVER Corporation
{judepark, nash}@jbnu.ac.kr
{donghyeon.jeon, seonhoon.kim, once.ihkang}@navercorp.com

Abstract

LM-BFF (Gao et al., 2021) achieves significant few-shot performance by using auto-generated prompts and adding demonstrations similar to an input example. To improve the approach of LM-BFF, this paper proposes **LM-BFF-MS**—**b**etter **f**ew-shot **f**ine-tuning of **l**anguage **m**odels with **m**ultiple **s**oft demonstrations by making its further extensions, which include 1) prompts with *multiple demonstrations* based on automatic generation of multiple label words; and 2) *soft demonstration memory* which consists of multiple sequences of *globally shared* word embeddings for a similar context. Experiments conducted on eight NLP tasks show that LM-BFF-MS leads to improvements over LM-BFF on five tasks, particularly achieving 94.0 and 90.4 on SST-2 and MRPC, respectively[1].

1 Introduction

The GPT-3 model (Brown et al., 2020) has achieved remarkable few-shot performance on natural language understanding tasks given a *natural language prompt* and $|K|$ labeled samples as *demonstrations* in the inputs without updating the model's weights. However, the GPT-3 model consists of 175B parameters, making it challenging to perform task-specific fine-tuning, which is often required in real-world applications.

To enable task-specific fine-tuning, *prompt-based few-shot fine-tuning* has been widely studied to encourage the few-shot capabilities of pre-trained language models (PLMs) equipped with label-specific *verbalizers* and *prompts* that are compatible with language models (Schick and Schütze, 2021a,b). Prompt-based fine-tuning reformulates downstream tasks as a masked language modeling problem, where a token (*label word*) is generated on a given prompt with a task-specific *template*. However, constructing optimal prompts requires domain expertise and the use of manual prompts can be suboptimal (Webson and Pavlick, 2021; Lu et al., 2021; Zhao et al., 2021).

Among the various methods of prompt-based fine-tuning, this study is based on the LM-BFF method (Gao et al., 2021), which uses a demonstration-aware prompt where a demonstration is produced by unmasking the example prompt in contexts similar to the input, inspired by the findings from the GPT-3 model (Brown et al., 2020). With demonstration-aware prompts, the LM-BFF outperforms the conventional fine-tuning approach and GPT-3's in-context learning. To improve LM-BFF, we propose **LM-BFF-MS**, **b**etter **f**ew-shot **f**ine-tuning of **l**anguage **m**odels with **m**ultiple **s**oft demonstration memory, based on the following two extensions:

1. **Prompts with multiple demonstrations**. While LM-BFF uses single demonstration[2], our model uses *multiple* demonstrations with different *label phrases*, where each demonstration is constructed per label phrase. Given that label phrases are semantically related or similar, it is expected that resulting demonstrations indirectly augment the vocabulary of the verbalizer with label phrases[3].

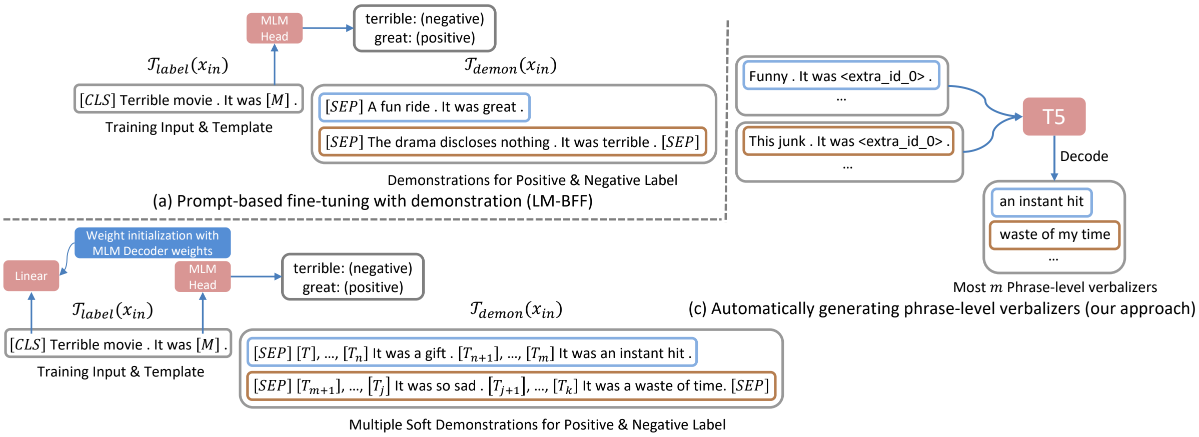
2. **Soft demonstration memory based on multiple sequences of word embeddings.** Unlike LM-BFF, which directly uses a sequence of hard tokens in the demonstration, inspired by the *soft prompts* of Lester et al. (2021), we replace them with a sequence of soft vectors as a proper context for each label phrase, where soft vectors are *globally shared soft examples* for each label phrase but are not sensitive to

[*] Corresponding author

[1] Our implementation is publicly available at https://github.com/judepark96/LM-BFF-MS

[2] Note that LM-BFF also explored sampling multiple demonstrations per label, but did not observe any improvement.

[3] Here, it is assumed that the set of label words is different from the set of label phrases.


MLM Head
terrible: (negative)
great: (positive)
$\mathcal{T}_{label}(x_{in})$
$\mathcal{T}_{demon}(x_{in})$
[CLS] Terrible movie . It was [M] .
Training Input & Template
[SEP] A fun ride . It was great .
[SEP] The drama discloses nothing . It was terrible . [SEP]
Demonstrations for Positive & Negative Label
Funny . It was <extra_id_0> .
...
This junk . It was <extra_id_0> .
...
T5
Decode
an instant hit
waste of my time
...
Most m Phrase-level verbalizers
Weight initialization with MLM Decoder weights
Linear
MLM Head
terrible: (negative)
great: (positive)
$\mathcal{T}_{label}(x_{in})$
$\mathcal{T}_{demon}(x_{in})$
[CLS] Terrible movie . It was [M] .
Training Input & Template
[SEP] [T], ..., [T_n] It was a gift . [T_{n+1}], ..., [T_m] It was an instant hit .
[SEP] [T_{m+1}], ..., [T_j] It was so sad . [T_{j+1}], ..., [T_k] It was a waste of time. [SEP]
Multiple Soft Demonstrations for Positive & Negative Label

(a) Prompt-based fine-tuning with demonstration (LM-BFF)

(c) Automatically generating phrase-level verbalizers (our approach)

(b) Prompt-based fine-tuning with multiple soft demonstration memory (our approach)

Figure 1: An illustration of (a) the prompt of LM-BFF (Prompt-based fine-tuning with demonstration), comparing to that of (b) our proposed LM-BFF-MS (Prompt-based fine-tuning with Multiple soft demonstration memory) in Section 3. The subfigure (c) shows the span-corrupted input and output of T5 used for automatic generation of phrase-level verbalizers as in Section 3.2. [M], [T_k], <extra_id_0>, blue and brown colored square box referred as mask and soft token, sentinel token of T5, positive, negative, respectively.

an input context. In our approach, soft vectors are considered as *automatically* generated demonstration that matches well for each label phrase, capturing the common context for the corresponding phrase. To train the soft demonstration memory effectively, we further introduce an auxiliary task, named *next demonstrations prediction* (NDP) task, inspired by NSP-BERT (Sun et al., 2021).

Following the previous setting of the LM-BFF, the experimental results on eight NLP datasets show that the proposed LM-BFF-MS leads to a better and more stable few-shot performance compared to the previous models. The contributions of this study are summarized as follows:

- We propose prompts with multiple soft demonstration memory based on the automatic generation of multiple label phrases and the use of soft demonstration memory that is armed with an auxiliary NDP task.

- We present promising results of the proposed method on eight NLP tasks by showing improved results on some datasets, particularly achieving state-of-the-art performance on SST-2 and MRPC.

2 Related Work

Prompt-based few-shot fine-tuning, which finetunes based on few-shot examples under a prompting setting, has been widely studied for moderately sized PLMs such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). For example, PET reformulates downstream tasks as a masked language modeling problem and performs gradient-based fine-tuning (Schick and Schütze, 2021a,b). AutoPrompt creates appropriate prompts for a set of discrete tokens using a gradient-guided search (Shin et al., 2020). *Null Prompts*—simple concatenations of the inputs and `[MASK]` token—achieve a free of prompt engineering (Logan et al., 2021). Instead of using hard prompts, there have also been works of using continuous vectors of prompt tokens, called *soft prompting*[4], including the work of Lester et al. (2021), which proposes *soft prompts* composed of learnable continuous embeddings while freezing the weight of PLMs; and Gu et al. (2021) proposes pre-training prompts by adding soft prompts into the pre-training stage to obtain a better initialization. The *demonstration-aware prompt* has also been explored by (Gao et al., 2021) with their proposed LM-BFF, where a demonstration is constructed by unmasking the masked prompt on a similar input example.

Unlike LM-BFF, which uses a single demonstration per label, our work uses '*multiple*' demonstrations that are provided for automatically generated label phrases. In addition, inspired by the method of soft prompting, we use '*soft*' demonstration memory based on globally shared soft vectors for prompt tokens, without using hard tokens of the similar context.

[4] Here, the soft prompting method refers to the methods of using unknown prompt-specific token embedding or hidden representations at prompt positions.

Model	**SST-2** (acc)	**MR** (acc)	**Subj** (acc)	**MRPC** (F1)
Majority†	50.9	50.0	50.0	81.2
Prompt-based zero-shot‡	83.6	80.8	51.4	61.9
“GPT-3” in-context learning	84.8 (1.3)	80.5 (1.7)	53.6 (0.8)	45.7 (6.0)
Fine-tuning	81.4 (3.8)	76.9 (5.9)	90.8 (1.8)	76.6 (2.5)
LM-BFF (man) + demonstration	92.6 (0.5)	86.6 (2.2)	92.3 (0.8)	77.8 (2.0)
DART	93.5 (0.5)	88.2 (1.0)	90.7 (1.4)	78.3 (4.5)
LM-BFF-MS	**94.0 (0.3)**	**88.3 (0.5)**	**92.7 (0.3)**	**80.4 (1.3)**
Fine-tuning (full)†	*95.0*	*90.8*	*97.0*	*91.4*

Model	**MNLI** (acc)	**SNLI** (acc)	**CR** (acc)	**MPQA** (F1)
Majority†	32.7	33.8	50.0	50.0
Prompt-based zero-shot‡	50.8	49.5	79.5	67.6
“GPT-3” in-context learning	52.0 (0.7)	47.1 (0.6)	87.4 (0.8)	63.8 (2.1)
Fine-tuning	45.8 (6.4)	48.4 (4.8)	75.8 (3.2)	72.0 (3.8)
LM-BFF (man) + demonstration	**70.7 (1.3)**	**79.7 (1.5)**	90.2 (1.2)	87.0 (1.1)
DART	67.5 (2.6)	75.8 (1.6)	**91.8 (0.5)**	-
LM-BFF-MS	68.2 (3.3)	73.9 (3.0)	90.8 (1.7)	**87.9 (0.4)**
Fine-tuning (full)†	*89.8*	*92.6*	*89.4*	*87.8*

Table 1: The main results with RoBERTa-large. †: the full training set is used. ‡: no training examples are used. Otherwise, we use $K = 16$ (# examples per class). The mean (and standard deviation) performance over five different splits is reported. Majority: majority class “GPT-3” in-context learning: using the in-context learning proposed in (Brown et al., 2020) with RoBERTa-large (no parameter updates); man: manual prompt; LM-BFF & DART: the performance in (Gao et al., 2021; Zhang et al., 2021) is reported. full: fine-tuning using full training set.

3 Fine-tuning with Multiple Soft Demonstration Memory

3.1 Background

Following on from (Gao et al., 2021), suppose that the input sentences $x_{in} = x_1$ and $x_{in} = (x_1, x_2)$ are presented for single-sentence and sentence-pair tasks, respectively. The template $\mathcal{T}$ is defined as $(\mathcal{T}_{label}, \mathcal{T}_{demon})$, where $\mathcal{T}_{label}$ is the template used to generate the *main* prompt for input x_{in} and $\mathcal{T}_{demon}$ is the additional template to generate demonstrations of input x_{in}. For example, $\mathcal{T}_{label}(x_{in})$ for a single sentence task is given as:

$$\mathcal{T}_{label}(x_{in}) = \texttt{[CLS]}\ x_1 \text{ It was } \texttt{[MASK]}\ .\ \texttt{[SEP]}$$

We use $\mathcal{T}_{label}$ with the manually designed templates of (Gao et al., 2021)[5].

To define $\mathcal{T}_{demon}$, suppose that $\mathcal{V}$ and $\mathcal{Y}$ are the vocabulary and label space, respectively. Let $\mathcal{M}_{wo}\colon \mathcal{Y} \to \mathcal{V}$ and $\mathcal{M}_{ph}^{(1)}, \cdots, \mathcal{M}_{ph}^{(m)}\colon \mathcal{Y} \to \mathcal{V}^*$ be mapping functions that convert a label into individual words and phrases, called *word*-level and *phrase*-level functions, respectively. For example, $\mathcal{M}_{wo}(pos)$ = “great”, $\mathcal{M}_{wo}(neg)$ = “terrible”, $\mathcal{M}_{ph}(pos)$ = “a gift”, $\mathcal{M}_{ph}(neg)$ = “a total waste of my time”. Let $\mathfrak{M}$ be the set of m phrase-level mapping functions, that is, $\mathfrak{M} = \{\mathcal{M}_{ph}^{(1)}, \cdots, \mathcal{M}_{ph}^{(m)}\}$. Given $\mathcal{M}_{ph} \in \mathfrak{M}$, $\tilde{\mathcal{T}}_{demon}(x_{in}, y, \mathcal{M}_{ph})$ is defined as the *unmasked* sequence of $\mathcal{T}_{label}(x_{in})$ by placing $\mathcal{M}_{ph}(y)$ instead of the `[MASK]` token; $\tilde{\mathcal{T}}_{demon}(x_{in}, y, \mathcal{M}_{ph})$ is obtained by first applying $\mathcal{T}_{label}$ to x_{in} to produce $\mathcal{T}_{label}(x_{in})$ and then replacing `[MASK]` with $\mathcal{M}_{ph}(y)$. For example, given $x_{in} = x_1, y = neg, \mathcal{M}_{ph}(neg)$ = “so sad”, $\tilde{\mathcal{T}}_{demon}(x_{in}, y, \mathcal{M}_{ph})$ is then obtained as: "x_1 It was so sad . `[SEP]`".

Now, suppose that $\mathcal{N}(x_{in}, y)$ is the set of training examples similar to x_{in} labeled with y. Then $\mathcal{T}_{demon}$ is defined as follows:

$$\mathcal{T}_{demon}(x_{in}) = \bigoplus_{\substack{\tilde{\mathcal{M}}_{ph} \in \mathfrak{M},\\ x \in \mathcal{N}(x_{in}, y),\\ y \in |\mathcal{Y}|}} \tilde{\mathcal{T}}_{demon}(x, y, \mathcal{M}_{ph}) \qquad (1)$$

where $\oplus$ denotes the concatenation operator.

Finally, x_{in} is converted to its prompted version $x_{prompt} = \mathcal{T}_{label}(x_{in}) \oplus \mathcal{T}_{demon}(x_{in})$, which is used as the input for the prompt-based few-shot fine-tuning.

Note that this setting includes the LM-BFF as a specific case with $|\mathcal{N}(x_{in}, y)| = 1$ per label and $\mathfrak{M} = \{\mathcal{M}_{wo}\}$ in Eq. (1), which refers to a *single* demonstration setting.

[5] We use Table 1 of (Gao et al., 2021) for $\mathcal{T}_{label}$ as described in Table 5. Note that we do not use auto-generated templates and label words.

3.2 Automatically Generating Phrase-Level Verbalizers

In contrast to the LM-BFF, we employ *phrase*-level mapping function, as it is theorized that it would enable a better representation of the demonstration than a word-level mapping function. The remaining part describes how to obtain the m phrase-level mapping functions in Eq. (1), $\mathcal{M}_{ph}^{(j)} \in \mathfrak{M}$.

To this end, we use T5 to generate label phrases using a properly designed span-corrupted input in the reverse manner of (Gao et al., 2021) which exploits T5 to automatically generate templates. The input for T5's encoder is merely the prompted sequence $\mathcal{T}_{label}(x_{in})$, however, with `[MASK]` as the span-corrupted token, the decoder then fills in the placeholders, removes duplicated results, and chooses the top m most likely generated sequences for phrase-level mapping functions of the corresponding label as described in Figure 1 (c). Our generation results are shown in Table 4.

3.3 Soft Demonstration Memory

Different from LM-BFF which explicitly finds similar training examples $\mathcal{N}(x_{in}, y)$, 'soft demonstration memory' is used, which consists of *globally shared soft examples* as demonstrations, assuming that each demonstration uses n *soft* tokens for a sentence as $\texttt{[}T_1\texttt{]} \cdots \texttt{[}T_n\texttt{]}$. Under a soft demonstration memory, $\mathcal{T}_{demon}(x_{in})$ is obtained using Eq. (1), but using the following definition of $\mathcal{N}(x_{in}, y)$[6]:

$$\mathcal{N}(x_{in}, y) = \left\{ \texttt{[}T_1^{(k)}\texttt{]} \cdots \texttt{[}T_n^{(k)}\texttt{]} \right\}_{k=1}^{m}$$

In single-sentence tasks, the soft demonstration memory maintains a total of $m \cdot |\mathcal{Y}|$ sentences each of which consists of n soft tokens, where the set of m sentences corresponds to each label. For example, when $n = 10$, $m = 5$, and $|\mathcal{Y}| = 2$, the total size of the global memory is 100.

To create m demonstrations, all examples of global memory are chosen without requiring a sample of similar examples. An illustration of the incorporated soft demonstration memory is described shown in Figure 1 (b).

3.4 Next Demonstration Prediction Task

To obtain a better representation of soft demonstration memory, we introduce the NDP task, which predicts whether positive (or negative) examples in $\mathcal{T}_{demon}(x_{in})$ are correctly matched with a positive (or negative) label word for the prompted input $\mathcal{T}_{label}(x_{in})$.

Dataset	Model	K=16
SST-2	Soft Prompting LM-BFF-MS	92.7 (0.5) **94.0 (0.3)**

Table 2: Few-shot performance comparison with *soft prompting* (Lester et al., 2021) and our approach.

To be more specific, the NDP task trains $P_{\texttt{NDP}}(y|x_{prompt}) = \texttt{softmax}(W_{\texttt{[MLM]}} h_{\texttt{[CLS]}} + b)$, where $W_{\texttt{[MLM]}} \in \mathbb{R}^{|\mathcal{Y}| \times d}$ are the output embedding weights of the label words in an MLM decoder. Finally, given a few-shot example (x_{in}, y), the training objective is defined as:

$$\mathcal{L} = \text{CE}\left(P(\texttt{[MASK]} = \mathcal{M}_{wo}(y)|x_{prompt})\right) + \lambda \cdot \text{CE}\left(P_{\texttt{NDP}}(y|x_{prompt})\right)$$

where CE is the cross-entropy loss function and λ is the hyper-parameter. Section 4.3 presents the effect of using the NDP loss compared to that without it.

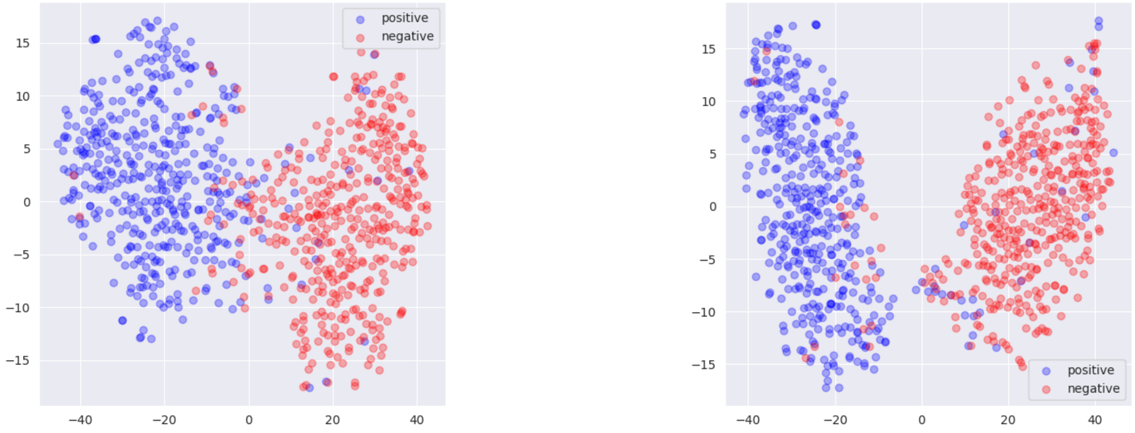
4 Experiments

The implementation details are provided in Appendix B. For a fair comparison, the same manual prompts for $\mathcal{T}_{label}$ in LM-BFF and LM-BFF-MS are used.

4.1 Main Results

As shown in Table 1, it is noticed that the proposed approach achieves a better and stable few-shot performance than the prior methods and the LM-BFF on five tasks. In particular, LM-BFF-MS achieves state-of-the-art performance on SST-2 and MRPC tasks, with 94.0 and 80.4, respectively. Moreover, it is observed that the performance variation of LM-BFF-MS are mostly lower than that of the prior methods except for the MNLI, SNLI, and CR tasks, implying that our approach is more stable than the existing models. On the other hand, LM-BFF-MS is weaker than LM-BFF on SNLI, although it shows comparable results to DART. We believe that the effect of global demonstration memory is task sensitive, suggesting that the *local* method of sampling similar demonstrations as in LM-BFF often needs to be employed for some tasks or specific input sentences.

[6] That is, the soft demonstration memory is a set of $|\mathcal{Y}|$ global memories each of which consists of m demonstrations.



(a) `[MASK]` w/o NDP Task

(b) `[MASK]` w/ NDP Task

Figure 2: A visualization of representation of `[MASK]` tokens.

4.2 Soft Demonstration Memory vs. Soft Prompting

To validate the use of soft demonstration memory, instead of inserting soft vectors into the demonstration parts. We further evaluate the soft prompting of (Lester et al., 2021) by prepending p soft vectors to the main template $\mathcal{T}_{label}(x_{in})$, where p is the length of the additional soft prompt[7].

Table 2 compares soft prompting with LM-BFF-MS on SST-2 and shows that LM-BFF-MS outperforms soft prompting under the setting of the same length of soft token. The results confirm that the gain of soft demonstration memory is not merely obtained by using additional parameters of soft vectors, but by effectively modeling the demonstration-aware context.

4.3 The Effect of Using Next Demonstration Prediction Task

Method	SST-2 (acc)
LM-BFF (man) + demonstration	92.6 (0.5)
DART	93.5 (0.5)
LM-BFF-MS	**94.0 (0.3)**
−next demonstration prediction	93.4 (0.4)

Table 3: Ablation study for NDP on SST-2 dataset.

To examine whether the use of the NDP task is indeed effective in LM-BFF-MS, Table 3 compares results of LM-BFF-MS with and without the NDP task on the SST-2 dataset. As shown in Table 3, LM-BFF-MS with NDP loss shows improved performance compared to that without NDP loss, providing positive evidence for our motivating hypothesis that the use of the NDP loss is helpful in enhancing the representation of soft demonstration memory.

To further analyze the effect of auxiliary NDP task, Figure 2 visualizes the representation of `[MASK]` tokens on SST-2 using t-SNE (van der Maaten and Hinton, 2008) compared, with and without the NDP task. As shown in Figure 2a and 2b, the representation learned using the NDP task is more discriminative than that learned without the NDP task, suggesting that the NDP task provides an effective additional loss for learning representations of soft demonstration memory.

5 Conclusion

This study proposed LM-BFF-MS—manual prompts with *multiple soft demonstration memory* based on the automatic generation of multiple label words and an auxiliary NDP task. Experiments showed that the proposed method outperforms prior works on five tasks: SST-2, MR, Subj, MRPC, and MPQA. Extending our work to a large soft demonstration memory and a combination of local and global memory is valuable for future investigations.

Acknowledgements

We would like to thank all anonymous reviewers for their valuable comments and suggestions.

[7] Here, p is fixed to be the same as the number of tokens used in the multiple demonstrations of the LM-BFF-MS.

References

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

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

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

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. PPT: pre-trained prompt tuning for few-shot learning. *CoRR*, abs/2109.04332.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 168–177, New York, NY, USA. Association for Computing Machinery.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *CoRR*, abs/2104.08691.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Robert Logan, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2021. Cutting down on prompts and parameters: Simple few-shot learning with language models.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *CoRR*, abs/2104.08786.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and

Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Yi Sun, Yu Zheng, Chao Hao, and Hangping Qiu. 2021. NSP-BERT: A prompt-based zero-shot learner through an original pre-training task-next sentence prediction. *CoRR*, abs/2109.03564.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Albert Webson and Ellie Pavlick. 2021. Do prompt-based models really understand the meaning of their prompts?

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3).

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Differentiable prompt makes pre-trained language models better few-shot learners. *CoRR*, abs/2108.13161.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

A Limitation

The main contribution of this work is the *multiple soft demonstration memory*, however, the number of available demonstrations is bounded by the maximum input length. Furthermore, unlike the GPT-3 model, the maximum input length of PLMs is usually 512, which is not sufficient to deal with more difficult tasks such as SNLI and MNLI. As shown in Table 1, despite the effectiveness of LM-BFF-MS, it shows a lower few-shot performance than previous studies on SNLI and MNLI. We believe that this is strongly related to automatic phrase-level generation and is bounded by the maximum input length. We leave this topic as a subject for future work.

B Implementation Details

B.1 Datasets & Setting

We used the following datasets—SNLI (Bowman et al., 2015), MNLI (Williams et al., 2018), SST-2 (Socher et al., 2013) MRPC (Dolan and Brockett, 2005), MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), MPQA (Wiebe et al., 2005), and Subj (Pang and Lee, 2004). This study followed the same experimental setting from LM-BFF (Gao et al., 2021).

B.2 Implementation

This proposed approach was implemented using PyTorch (Paszke et al., 2019) and HuggingFace Transformers (Wolf et al., 2020). Experiments were conducted with Nvidia Quadro RTX 8000 GPU. All optimizations were performed using the AdamW optimizer with a linear warm-up of the learning rate. The warmup proportion is 0.6. The gradients are clipped if their norms exceed 1.0.

A T5-large and beam search (e.g., beam width: 30) were used to generate phrase-level verbalizers automatically in a zero-shot manner.

B.3 Multiple Soft Demonstration Memory Setting

SST-2, MR, CR, Subj, MRPC, MPQA

- Length of soft tokens: $n = 10$
- Number of target label: $|\mathcal{Y}| = 2$
- Demonstrations per label: $m = 5$
- Total: $|T| = n \cdot m \cdot |\mathcal{Y}| = 100$

MNLI

- Length of soft tokens: $n = 20$
- Number of target label: $|\mathcal{Y}| = 3$

Task	label	Phrase-level Verbalizers
SST-2	negative	[well worth the effort, a total waste of my time, a real treat to watch, so sad, a cultural revolution]
	positive	[an instant hit, a gift, entertaining on an inferior level, an unforgettable experience, a thriller with an edge]
MR	negative	[delicious, time, ms, the right thing to do, a very sad movie]
	positive	[refreshing, the best film of the year, well worth the money, such a shame, released on friday]
Subj	subjective	[not a documentary, godard at his best, a funny film, not a great movie, not one of them]
	objective	[the story of dr, not an easy task, a great site, not a film to be missed, not a great film]
MRPC	not_equivalent	[For the three-month daily average, According to the Washington Post, This is not unanticipated, At midday Monday, ? In the 1990s]
	equivalent	[On Friday, Yesterday, JERUSALEM, Hi, Today]
MNLI	contradiction	[In an interview with CNN, we hope, California Rural Justice Consortium, Realigning, In this simulation]
	entailment	[For more information, He's nice, Ueno, I got good results, Exercise Bicycle]
	neutral	[At the University of Georgia, million, Firstly, Arthur Schlesinger, I was embarrassed]
SNLI	contradiction	[Car in garage, Florida Marlins, The pipe is black, In the boat, , The man is painting.]
	entailment	[Uncle Henry, At a trailer, Hippie is walking on foot, On a dusty path, In this kitchen]
	neutral	[According to locals, Playing with a ball, ", A group of people are walking", Mountains in the background]
CR	negative	[a complete waste of time, working fine for me, not working on my other phone, supposed to work, the same for me]
	positive	[exactly what i was looking for, a lot of fun to use, a great day, a pleasure to work with you, the perfect phone for me]
MPQA	negative	[ful, here, China, good, customers]
	positive	[trade, know , transparent, -tuned, and values]

Table 4: Automatic generation for phrase-level verbalizers $\mathcal{M}_{ph}$ used in our experiments.

Task	Template	Label words
SST-2	It was `[MASK]` .	positive: great, negative: terrible
MR	It was `[MASK]` .	positive: great, negative: terrible
CR	It was `[MASK]` .	positive: great, negative: terrible
MPQA	It was `[MASK]` .	positive: great, negative: terrible
Subj	This is `[MASK]` .	subjective: subjective, objective: objective
MNLI	? `[MASK]` ,	entailment: Yes, netural: Maybe, contradiction: No
SNLI	? `[MASK]` ,	entailment: Yes, netural: Maybe, contradiction: No
MRPC	? `[MASK]` ,	equivalent: Yes, not_equivalent: No

Table 5: Manual templates and label words $\mathcal{M}_{wo}$ that we used in our experiments from LM-BFF (Gao et al., 2021).

- Demonstrations per label: $m = 1$
- Total: $|T| = n \cdot m \cdot |\mathcal{Y}| = 60$

SNLI

- Length of soft tokens: $n = 10$
- Number of target label: $|\mathcal{Y}| = 3$
- Demonstrations per label: $m = 1$
- Total: $|T| = n \cdot m \cdot |\mathcal{Y}| = 30$

B.4 Training Example

Suppose that we train SST-2 dataset following setting: $n = 2, |\mathcal{Y}| = 2$, and $m = 2$. Then x_{prompt} is formed as follows:

$$\begin{aligned} x_{prompt} = {} & \texttt{[CLS]}\ x_1 \text{ It was } \texttt{[MASK]}\ .\ \texttt{[SEP]} \\ & \texttt{[}T_1\texttt{]}\ \texttt{[}T_2\texttt{]} \text{ It was an instant hit} \\ & \texttt{[}T_3\texttt{]}\ \texttt{[}T_4\texttt{]} \text{ It was a gift } \texttt{[SEP]} \\ & \texttt{[}T_5\texttt{]}\ \texttt{[}T_6\texttt{]} \text{ It was well worth the effort} \\ & \texttt{[}T_7\texttt{]}\ \texttt{[}T_8\texttt{]} \text{ It was a total waste of my time } \texttt{[SEP]} \end{aligned}$$

where x_1, $\texttt{[}T_1\texttt{]}, \cdots \texttt{[}T_4\texttt{]}$, $\texttt{[}T_5\texttt{]}, \cdots \texttt{[}T_8\texttt{]}$ are the input sentence and multiple soft demonstration memory for positive and negative labels, respectively. In this case, $W_{\texttt{[MLM]}} \in \mathbb{R}^{2 \times d}$ is the output embedding weights of label words (e.g., positive: 'great', negative: 'terrible') in a MLM Decoder for the NDP task.

Towards Fair Evaluation of Dialogue State Tracking by Flexible Incorporation of Turn-level Performances

Suvodip Dey, Ramamohan Kummara, Maunendra Sankar Desarkar
Indian Institute of Technology Hyderabad, India
cs19resch01003@iith.ac.in, cs19mds11004@iith.ac.in, maunendra@cse.iith.ac.in

Abstract

Dialogue State Tracking (DST) is primarily evaluated using Joint Goal Accuracy (JGA) defined as the fraction of turns where the ground-truth dialogue state exactly matches the prediction. Generally in DST, the dialogue state or belief state for a given turn contains all the intents shown by the user till that turn. Due to this cumulative nature of the belief state, it is difficult to get a correct prediction once a misprediction has occurred. Thus, although being a useful metric, it can be harsh at times and underestimate the true potential of a DST model. Moreover, an improvement in JGA can sometimes decrease the performance of turn-level or non-cumulative belief state prediction due to inconsistency in annotations. So, using JGA as the only metric for model selection may not be ideal for all scenarios. In this work, we discuss various evaluation metrics used for DST along with their shortcomings. To address the existing issues, we propose a new evaluation metric named **F**lexible **G**oal **A**ccuracy (**FGA**). FGA is a generalized version of JGA. But unlike JGA, it tries to give penalized rewards to mispredictions that are locally correct i.e. the root cause of the error is an earlier turn. By doing so, FGA considers the performance of both cumulative and turn-level prediction flexibly and provides a better insight than the existing metrics. We also show that FGA is a better discriminator of DST model performance.

1 Introduction

Dialogue State Tracking (DST) is at the core of task-oriented dialogue systems. It is responsible for keeping track of the key information exchanged during a conversation. With the growing popularity of task-based conversational agents, it is essential to review the evaluation of DST to appropriately measure the progress in this evolving area.

The task of DST is to predict the user intent through dialogue states (Henderson et al., 2014). Fig. 1 shows an example DST task from MultiWOZ (Budzianowski et al., 2018) dataset. Let U_t and S_t be the user and system utterances respectively at turn t. Then a typical conversation can be expressed as $D = \{U_0, (S_1, U_1), ...(S_n, U_n)\}$. The commonly used ground-truth dialogue state for DST is the belief state. Belief state B_t for turn t is defined as the set of *(domain, slot, slot-value)* triplets that have been extracted till turn t, thereby it is cumulative in nature. The objective of DST is to predict B_t given the dialogue history till turn t.

The primary metric for evaluating DST is Joint Goal Accuracy (JGA). It compares the predicted dialogue states to the ground truth B_t at each dialogue turn t (Henderson et al., 2014). As the belief state is cumulative, it is very unlikely for a model to get back a correct prediction after a misprediction. This is why it can provide an underestimated performance in certain cases. Besides, JGA completely ignores the performance of turn-specific local predictions. Let T_t be the turn-level belief state that contains all the intents or *(domain, slot, slot-value)* triplets expressed by the user only at turn t. Ideally, a model with higher JGA should also perform equally well to predict T_t. But, we observe that improving JGA can sometimes degrade the performance of predicting T_t mainly due to the presence of annotation inconsistencies in the available datasets. For example, in Fig. 1, the presence of *(hotel, area, centre)* and absence of *(attraction, name, all saints church)* in ground-truth B_2 and B_4 shows such inconsistencies. So, the generalization of the model may get compromised if the model selection is done only using JGA. Annotation inconsistencies and errors are common in real-world datasets. Hence, to provide a fair estimate, it requires not only track the performance of the cumulative belief state but also turn-level belief state as well.

In this work, we address these issues of JGA by proposing a novel evaluation metric for DST called **F**lexible **G**oal **A**ccuracy (**FGA**). The central idea of

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 318 - 324
May 22-27, 2022 ©2022 Association for Computational Linguistics

FGA is to partially penalize a misprediction which is locally correct i.e. the source of the misprediction is some earlier turn. The main contributions of our work are as follows [1]:

- Detailed analysis of the existing DST metrics.
- Proposal of Flexible Goal Accuracy (FGA) than can keep track of both joint and turn-level performances simultaneously.
- Justification of FGA along with performance comparison on the MultiWOZ dataset.

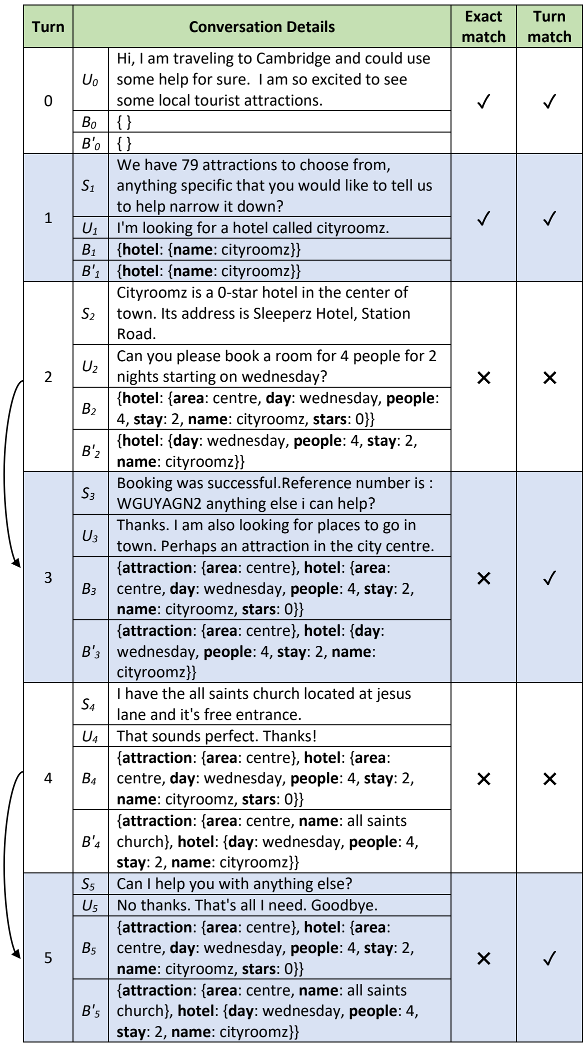
2 Discussion on existing DST metrics

2.1 Joint goal accuracy

Joint accuracy or joint goal accuracy (JGA) checks whether the set of predicted belief states exactly matches the ground truth for a given user turn (Henderson et al., 2014; Wu et al., 2019). Let B_t and B'_t be the set of ground-truth and predicted belief states at turn t. Then the prediction of turn t is considered to be correct if and only if B_t exactly matches B'_t. Fig. 1 shows an illustration of the predicted belief state where the predictions of B'_t are generated using SOM-DST (Kim et al., 2020). In the example, there are 2 out of 6 correct predictions of B'_t that result in a JGA score of 33.33% for the whole conversation.

Although joint goal accuracy is a convenient metric to evaluate DST, it has certain limitations. The main source of the issue is the cumulative nature of ground-truth B_t. As a result, once a misprediction has occurred, it is difficult to get back a correct prediction in subsequent turns. For example, in Fig. 1, the prediction goes wrong in Turn 2 which affects all the later predictions. So, it is very likely to get a JGA of zero if the model somehow mispredicts the first turn. Therefore, JGA can undermine the true potential of a DST model and provide an underestimated performance.

In addition, JGA does not take into account turn-level performances. For instance, in Fig. 1, Turn 3 and 5 are locally correct but JGA will mark them 0 since B_t and B'_t has not matched exactly. Normally, it is expected that increasing the exact matches will also reflect in turn-level matches. But we observed that sometimes increasing exact matches can decrease turn-level matches mainly due to annotation inconsistencies. So, one should be careful while using only joint accuracy for model selection. Besides, the available DST datasets (like MultiWOZ) contain a lot of annotation errors (Zang et al., 2020). For example in turn 4, the model has predicted the intent *(attraction, name, all saints church)*. Although the prediction looks rational, the triplet is absent in the ground-truth. So, if a mismatch occurs due to an annotation error, it is highly probable that all the subsequent turns will be marked incorrect leading to an underestimated performance.



Turn		Conversation Details	Exact match	Turn match
0	U_0	Hi, I am traveling to Cambridge and could use some help for sure. I am so excited to see some local tourist attractions.	✓	✓
	B_0	{ }		
	B'_0	{ }		
1	S_1	We have 79 attractions to choose from, anything specific that you would like to tell us to help narrow it down?	✓	✓
	U_1	I'm looking for a hotel called cityroomz.		
	B_1	{**hotel**: {**name**: cityroomz}}		
	B'_1	{**hotel**: {**name**: cityroomz}}		
2	S_2	Cityroomz is a 0-star hotel in the center of town. Its address is Sleeperz Hotel, Station Road.	×	×
	U_2	Can you please book a room for 4 people for 2 nights starting on wednesday?		
	B_2	{**hotel**: {**area**: centre, **day**: wednesday, **people**: 4, **stay**: 2, **name**: cityroomz, **stars**: 0}}		
	B'_2	{**hotel**: {**day**: wednesday, **people**: 4, **stay**: 2, **name**: cityroomz}}		
3	S_3	Booking was successful.Reference number is : WGUYAGN2 anything else i can help?	×	✓
	U_3	Thanks. I am also looking for places to go in town. Perhaps an attraction in the city centre.		
	B_3	{**attraction**: {**area**: centre}, **hotel**: {**area**: centre, **day**: wednesday, **people**: 4, **stay**: 2, **name**: cityroomz, **stars**: 0}}		
	B'_3	{**attraction**: {**area**: centre}, **hotel**: {**day**: wednesday, **people**: 4, **stay**: 2, **name**: cityroomz}}		
4	S_4	I have the all saints church located at jesus lane and it's free entrance.	×	×
	U_4	That sounds perfect. Thanks!		
	B_4	{**attraction**: {**area**: centre}, **hotel**: {**area**: centre, **day**: wednesday, **people**: 4, **stay**: 2, **name**: cityroomz, **stars**: 0}}		
	B'_4	{**attraction**: {**area**: centre, **name**: all saints church}, **hotel**: {**day**: wednesday, **people**: 4, **stay**: 2, **name**: cityroomz}}		
5	S_5	Can I help you with anything else?	×	✓
	U_5	No thanks. That's all I need. Goodbye.		
	B_5	{**attraction**: {**area**: centre}, **hotel**: {**area**: centre, **day**: wednesday, **people**: 4, **stay**: 2, **name**: cityroomz, **stars**: 0}}		
	B'_5	{**attraction**: {**area**: centre, **name**: all saints church}, **hotel**: {**day**: wednesday, **people**: 4, **stay**: 2, **name**: cityroomz}}		

Figure 1: Illustration of DST task. "Exact Match" compares Ground truth belief state B_t and Predicted belief state B'_t. "Turn Match" indicates the correctness of turn-level non-cumulative belief state prediction. Arrows represent the propagation of errors.

Hence, using joint goal accuracy for evaluating DST works fine if there are no annotation errors and the sole purpose is to improve the prediction of cumulative belief state. Otherwise, there is a need to include turn-level performance in order to obtain a fair evaluation of a DST model.

[1] Code is available at github.com/SuvodipDey/FGA

2.2 Slot Accuracy

Slot accuracy (SA) is a relaxed version of JGA that compares each predicted *(domain, slot, slot-value)* triplet to its ground-truth label individually (Wu et al., 2019). Let S be the set of unique domain-slot pairs in the dataset. Let B_t and B'_t be the set of ground-truth and predicted belief states respectively. Then slot accuracy at turn t is defined as

$$SA = \frac{|S| - |X| - |Y| + |P \cap Q|}{|S|}, \qquad (1)$$

where $X = (B_t \setminus B'_t)$, $Y = (B'_t \setminus B_t)$, P is the set of unique domain-slot pairs from X, and Q is the set of unique domain-slot pairs from Y. Basically, in Equation 1, $|X|$ and $|Y|$ represent the number of false negatives and false positives respectively. Note that if the value of a ground-truth domain-slot pair is wrongly predicted then this misprediction will be counted twice (once in both X and Y). The term $|P \cap Q|$ in the above equation helps to rectify this overcounting. In MultiWOZ, the value of $|S|$ is 30. For Turn 2 in our running example, since $|B_1 \setminus B'_1| = 2$ and $|B'_1 \setminus B_1| = 0$, slot accuracy is equal to $\frac{(30-2-0-0)}{30}$ i.e. 93.33%. Slot accuracy for the entire conversation in Fig. 1 is 94.44%.

The value of slot accuracy can be very misleading. For instance, even if the prediction of Turn 2 is wrong in Fig. 1, we get a slot accuracy of 93.33% which is extremely high. Basically, slot accuracy overestimates the DST performance. Let us exhibit this fact by considering the case where we predict nothing for all turns i.e. $B'_t = \emptyset, \forall t$. Then, slot accuracy simplifies to $\frac{|S|-|B_t|}{|S|}$. It is natural that $|B_t| << |S|$ because a conversation will typically have only a small number of domain-slot pairs *live* at any time. As a result, slot accuracy remains on the higher side ($\approx$ 81% for MultiWOZ 2.1) even if we predict nothing. For datasets with a larger number of domain/slots, since $|S|$ is large, slot accuracy will be close to 1 for almost all scenarios. Thus, slot accuracy is a poor metric to evaluate DST.

2.3 Average Goal accuracy

Average goal accuracy (AGA) is a relatively newer metric proposed to evaluate the SGD dataset (Rastogi et al., 2020). Here, the slots that have a non-empty assignment in the ground-truth dialogue state are only considered during evaluation. Let $N_t \subseteq B_t$ be the set of ground-truth triplets having non-empty slot-values. Then AGA is computed as $\frac{|N_t \cap B'_t|}{|N_t|}$ where B'_t is the predicted belief state for turn t. The turns having $N_t = \emptyset$ are ignored during the computation of AGA. In Fig. 1, AGA for turn 2 is 4/6, and 76.19% for the entire conversation.

This metric has mainly two limitations. Firstly, AGA is only recall-oriented and thereby does not consider the false positives. Ignoring the false positives makes this metric insensitive to extraneous triplets in the predicted belief state. However, this issue can be easily addressed by redefining AGA as $\frac{|N_t \cap B'_t|}{|N_t \cup B'_t|}$. But there still exists a second major problem with AGA. Note that even if a turn is completely wrong, AGA for that turn can still be higher because of the correct predictions in the previous turns. For example, even if turn 2 and 4 are incorrect, we get an AGA of 4/6 and 5/7 respectively which clearly indicates an overestimation.

3 Flexible Goal Accuracy

From the previous discussion, it is evident that despite a few limitations, joint goal accuracy is superior to the other two metrics. This is why with the objective to obtain a better evaluation metric for DST, we address the shortcomings of JGA by proposing a new metric called Flexible goal accuracy (FGA). The description of FGA is presented in the next part of this section, whereas its working is described as a pseudo-code in Algo. 1.

For a given a turn t, an error in belief state prediction (i.e. $B_t \neq B'_t$) can occur in two ways: 1) the source of the error is turn t itself i.e. the turn-level prediction is wrong, 2) the turn-level prediction of turn t is correct but the source of the error is some earlier turn $t_{err} \prec t$. FGA works differently from JGA only for type 2 errors. Unlike JGA, FGA does not penalize type 2 errors completely. It assigns a penalized score based on the distance between the error turn (t_{err}) and the current turn (t) and the penalty is inversely proportional to this distance ($t - t_{err}$). The main idea is to forget the mistakes with time in order to attain a fair judgment of a DST model offline.

We decide the correctness of a turn-level match using the logic shown in line 10 of Algo. 1. A turn $t > 0$ is locally correct if ($T'_t \subseteq B_t$ and $T_t \subseteq B'_t$) where $T_t = B_t \setminus B_{t-1}$ and $T'_t = B'_t \setminus B'_{t-1}$. In other words, a turn-level or local match indicates that all the intents shown by the user in a particular turn have been correctly detected without any false positives. Just comparing T_t and T'_t to check a turn-level or local match can be erroneous because it will not credit the model for error corrections.

```
Algorithm 1: FGA for single conversation
Input: $B$ = list of groun-truth belief states,
       $B'$ = list of predicted belief states,
       $N$ = #turns
Output: Flexible goal accuracy
1  $T = \{0, 1, \ldots, N-1\}$ , $t_{err} \leftarrow -\infty$, f = 0
2  for $t \in T$ do
3      $w \leftarrow 1$
4      if $B_t \neq B'_t$ then
5          if $t = 0$ then
               /* Type 1 error        */
6              $w \leftarrow 0$ , $t_{err} \leftarrow t$
7          else
8              $T_t \leftarrow B_t \setminus B_{t-1}$
9              $T'_t \leftarrow B'_t \setminus B'_{t-1}$
10             if $T'_t \not\subseteq B_t$ or $T_t \not\subseteq B'_t$ then
                   /* Type 1 error    */
11                 $w \leftarrow 0, t_{err} \leftarrow t$
12             else
                   /* Type 2 error    */
13                 $x \leftarrow (t - t_{err})$
14                 $w \leftarrow 1 - \exp(-\lambda x)$
15     $f \leftarrow f + w$
16 return $f/N$
```

For the penalty function, we use the CDF of exponential distribution (shown in Line 14 of Algo. 1) parameterized by λ where $\lambda \geq 0$. Clearly, the strictness of FGA is inversely proportional to λ. Note that $\lambda = 0$ will reduce FGA to JGA (strict metric) whereas $\lambda \to \infty$ will report only the accuracy on turn-level matches (relaxed metric). Finding the appropriate λ for a specific DST task should be done carefully in order to match the desired evaluation criteria. However, we can take a theoretical stand and approximate the hyper-parameter value as $\lambda = -ln(1-p)/t_f$ where t_f is the number of turns that it will take to forget a mistake by factor p where $(0 \leq p < 1)$. For example, if t_f=6 and p=0.95, then λ=0.499. So, the strictness of FGA is directly proportional to t_f and inversely proportional to p. If the dataset is clean, one can alternatively find the best λ through a human evaluation, although it would require additional human effort. Hence, we can flexibly set the strictness criteria of FGA through the hyper-parameter λ according to our requirement.

In our running example (Fig. 1), the FGA score for each turn with $\lambda = 0.5$ is $\{1, 1, 0, 0.39, 0, 0.39\}$ which results in a FGA score of 46.33% for the entire conversation. We can observe two things from these numbers. Firstly, it is not overestimating in comparison to SA and AGA. Secondly, it gives a better estimate than JGA in keeping track of both exact and turn-level matches simultaneously. Hence, FGA can provide a relatively balanced estimate than the existing metrics even in the presence of annotation errors and inconsistencies.

4 Result and Analysis

In this section, we report the performance of FGA along with the other metrics on four different DST models: TRADE (Wu et al., 2019), Hi-DST (Dey and Desarkar, 2021), SOM-DST (Kim et al., 2020), and Trippy (Heck et al., 2020). We use the MultiWOZ 2.1 dataset (Eric et al., 2020) as most of the recent progress in DST are showcased on this dataset. The results are reported in Table 1. Since the MultiWOZ dataset covers many domains (hotel, restaurant, taxi, train, attraction) where each domain may have different levels of tolerance (intuitively train, taxi booking may be strict whereas information seeking about attraction, restaurant domains may be lenient), an overall common/single strictness setting for the entire dataset may be difficult to reach at. Hence, we reported the FGA score for multiple values of hyper-parameter λ rather than showing the result for a single value. For the same reason, we did not try to find the best λ for evaluating the MultiWOZ dataset.

From Table 1, we can observe that Trippy has the best JGA. Currently, most of the state-of-the-art DST performances are shown using Trippy. However, we can notice that Trippy does not have the same performance gain for turn-level matches. It has lesser turn-level matches than SOM-DST and Hi-DST. This behavior of Trippy can be a side-effect of boosting the JGA using its intricate featurization. In contrast, Hi-DST optimizes explicitly for turn-level non-cumulative belief states, thereby achieving better turn-level accuracy at the expense of JGA. Among the four models, SOM-DST performs well for both objectives because of their sophisticated selective overwrite mechanism. Now, by comparing the numbers of Table 1, we can infer that FGA does a better job in providing a fair estimate while considering both exact and turn-level matches. Moreover, we can also notice that FGA acts as a better discriminator of DST models in comparison to the existing metrics.

Human Evaluation: We conducted a human

Model	#Turns	#M1	#M2	JGA	SA	AGA	FGA$_{0.25}$	FGA$_{0.5}$	FGA$_{0.75}$	FGA$_{1}$
TRADE	7368	3600	5287	48.86%	96.96%	88.79%	56.58%	61.19%	64.16%	66.18%
Hi-DST	7368	3622	5903	49.16%	96.70%	90.74%	61.31%	67.69%	71.47%	73.91%
SOM-DST	7368	3912	6084	53.09%	97.36%	91.71%	64.94%	71.04%	74.61%	76.88%
Trippy	7368	3926	5875	53.28%	97.30%	90.75%	63.24%	68.67%	71.97%	74.13%

Table 1: Comparison of DST metrics. “M1” and “M2” represents exact and turn-level matches respectively. “FGA$_\text{x}$” indicates the FGA value calcualated using λ=x.

evaluation involving 11 evaluators on 100 randomly picked conversations from the MultiWOZ 2.1 test data. For each turn in a conversation, we provided the system and user utterances along with the ground-truth and predicted belief states. The predictions were generated using SOM-DST. For each conversation, the evaluators were asked to report their satisfaction (1) or dissatisfaction (0) with the performance of the model in keeping track of user intent throughout the conversation. Pearson correlation coefficient of JGA and FGA (with $\lambda = 0.5$) with human ratings came out to be 0.33 and 0.37 respectively. This shows that FGA is slightly better correlated than JGA with human evaluation.

5 Conclusion

In this work, we analyzed the limitations of existing DST metrics. We argued that joint accuracy can underestimate the power of a DST algorithm, whereas slot and average goal accuracy can overestimate it. We addressed the issues of joint accuracy by introducing Flexible goal accuracy (FGA) which tries to give partial credit to mispredictions that are locally correct. We justified that FGA provides a relatively balanced estimation of DST performance along with better discrimination property. In conclusion, FGA is a practical and insightful metric that can be useful to evaluate future DST models.

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Suvodip Dey and Maunendra Sankar Desarkar. 2021. Hi-DST: A hierarchical approach for scalable and extensible dialogue state tracking. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 218–227, Singapore and Online. Association for Computational Linguistics.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 422–428, Marseille, France. European Language Resources Association.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. TripPy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 35–44, 1st virtual meeting. Association for Computational Linguistics.

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014. The second dialog state tracking challenge. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 263–272, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 567–582, Online. Association for Computational Linguistics.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:8689–8696.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819, Florence, Italy. Association for Computational Linguistics.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2 : A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117, Online. Association for Computational Linguistics.

A Appendix

A.1 MultiWOZ Dataset

MultiWOZ (Budzianowski et al., 2018) is a popular DST corpus that contains both single and multi-domain conversations. For this work, we used MultiWOZ 2.1 (Eric et al., 2020) which is an updated version of the original MultiWOZ 2.0 dataset. In addition to the original dataset, MultiWOZ 2.1 contains fixes to some noisy annotations. Table 2 shows few elementary statistics of the dataset.

Data	**#Conversations**	**#Turns**	**Avg. turns**
Train	8420	56668	6.73
Dev	1000	7374	7.37
Test	999	7368	7.37

Table 2: Elementary statistics of MultiWOZ 2.1 dataset. “Avg. turns” indicate average turns per conversation.

A.2 Result generation procedure

We generated results for four DST models - Trade (Wu et al., 2019) [2], Hi-DST (Dey and Desarkar, 2021) [3], SOM-DST (Kim et al., 2020) [4], and Trippy (Heck et al., 2020) [5]. We used their official code to train them on MutiWOZ 2.1 dataset. All four models generate an inference file that contains the predicted belief states for the test set. We used these inference files to compute the values of different metrics shown in Table 1. As we trained all the models from scratch, the results may not be exactly the same as those reported in the original paper.

A.3 Human evaluation format

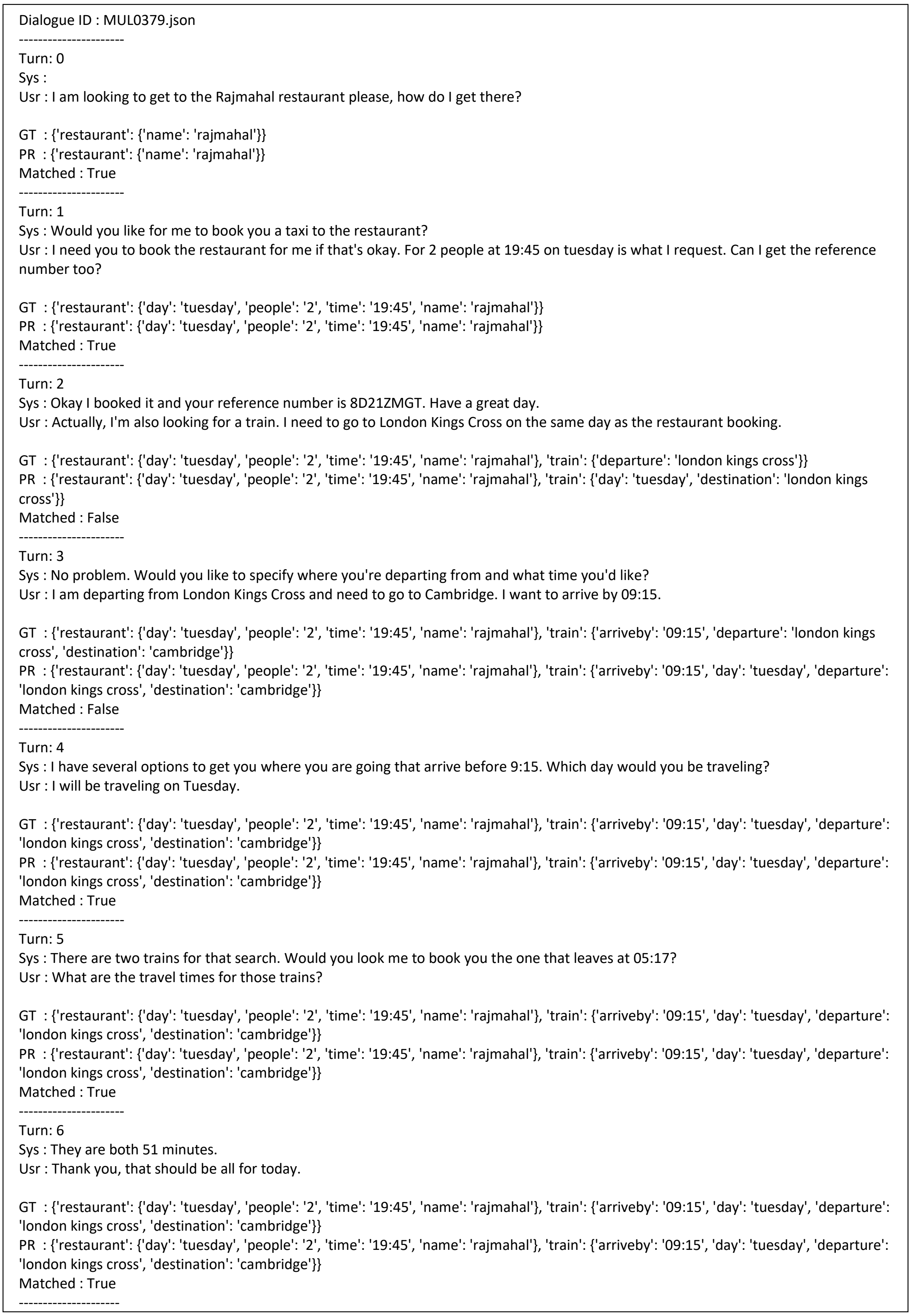
For each randomly picked conversation for human evaluation, we prepared a file that logged the utterances, ground-truth, and predicted belief state for each turn. Additionally, we indicated whether the ground truth exactly matched the predicted belief state to speed up the evaluation process. A sample file format is shown in Fig. 2.

[2] github.com/jasonwu0731/trade-dst

[3] github.com/SuvodipDey/Hi-DST

[4] github.com/clovaai/som-dst

[5] gitlab.cs.uni-duesseldorf.de/general/dsml/trippy-public



```
Dialogue ID : MUL0379.json
----------------------
Turn: 0
Sys :
Usr : I am looking to get to the Rajmahal restaurant please, how do I get there?

GT  : {'restaurant': {'name': 'rajmahal'}}
PR  : {'restaurant': {'name': 'rajmahal'}}
Matched : True
----------------------
Turn: 1
Sys : Would you like for me to book you a taxi to the restaurant?
Usr : I need you to book the restaurant for me if that's okay. For 2 people at 19:45 on tuesday is what I request. Can I get the reference
number too?

GT  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}}
PR  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}}
Matched : True
----------------------
Turn: 2
Sys : Okay I booked it and your reference number is 8D21ZMGT. Have a great day.
Usr : Actually, I'm also looking for a train. I need to go to London Kings Cross on the same day as the restaurant booking.

GT  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'departure': 'london kings cross'}}
PR  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'day': 'tuesday', 'destination': 'london kings
cross'}}
Matched : False
----------------------
Turn: 3
Sys : No problem. Would you like to specify where you're departing from and what time you'd like?
Usr : I am departing from London Kings Cross and need to go to Cambridge. I want to arrive by 09:15.

GT  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'arriveby': '09:15', 'departure': 'london kings
cross', 'destination': 'cambridge'}}
PR  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'arriveby': '09:15', 'day': 'tuesday', 'departure':
'london kings cross', 'destination': 'cambridge'}}
Matched : False
----------------------
Turn: 4
Sys : I have several options to get you where you are going that arrive before 9:15. Which day would you be traveling?
Usr : I will be traveling on Tuesday.

GT  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'arriveby': '09:15', 'day': 'tuesday', 'departure':
'london kings cross', 'destination': 'cambridge'}}
PR  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'arriveby': '09:15', 'day': 'tuesday', 'departure':
'london kings cross', 'destination': 'cambridge'}}
Matched : True
----------------------
Turn: 5
Sys : There are two trains for that search. Would you look me to book you the one that leaves at 05:17?
Usr : What are the travel times for those trains?

GT  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'arriveby': '09:15', 'day': 'tuesday', 'departure':
'london kings cross', 'destination': 'cambridge'}}
PR  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'arriveby': '09:15', 'day': 'tuesday', 'departure':
'london kings cross', 'destination': 'cambridge'}}
Matched : True
----------------------
Turn: 6
Sys : They are both 51 minutes.
Usr : Thank you, that should be all for today.

GT  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'arriveby': '09:15', 'day': 'tuesday', 'departure':
'london kings cross', 'destination': 'cambridge'}}
PR  : {'restaurant': {'day': 'tuesday', 'people': '2', 'time': '19:45', 'name': 'rajmahal'}, 'train': {'arriveby': '09:15', 'day': 'tuesday', 'departure':
'london kings cross', 'destination': 'cambridge'}}
Matched : True
---------------------
```

Figure 2: Data format for human evaluation

Exploiting Language Model Prompts Using Similarity Measures: A Case Study on the Word-in-Context Task

Mohsen Tabasi[1], **Kiamehr Rezaee**[2] and **Mohammad Taher Pilehvar**[3]

[1]Department of CE, Iran University of Science and Technology, Tehran, Iran
[2*]Cardiff NLP, School of Computer Science and Informatics, Cardiff University, UK
[3]Tehran Institute for Advanced Studies, Khatam University, Tehran, Iran
s_tabasi@comp.iust.ac.ir, rezaee.k@cardiff.ac.uk
mp792@cam.ac.uk

Abstract

As a recent development in few-shot learning, prompt-based techniques have demonstrated promising potential in a variety of natural language processing tasks. However, despite proving competitive on most tasks in the GLUE and SuperGLUE benchmarks, existing prompt-based techniques fail on the semantic distinction task of the Word-in-Context (WiC) dataset. Specifically, none of the existing few-shot approaches (including the in-context learning of GPT-3) can attain a performance that is meaningfully different from the random baseline. Trying to fill this gap, we propose a new prompting technique, based on similarity metrics, which boosts few-shot performance to the level of fully supervised methods. Our simple adaptation shows that the failure of existing prompt-based techniques in semantic distinction is due to their improper configuration, rather than lack of relevant knowledge in the representations. We also show that this approach can be effectively extended to other downstream tasks for which a single prompt is sufficient.[†]

1 Introduction

Recently, there has been a resurgence of interest in few-shot learning, especially after the introduction of GPT-3 (Brown et al., 2020). The current dominant few-shot approach is the so-called prompt-based learning which involves a simple reformulation of the target task as a cloze-style (Taylor, 1953) fill-in-the-blank objective. The core idea is to extract knowledge by asking the right question from the pre-trained language model (PLM) using a task-specific prompting template which directs the PLM to generate a textual output corresponding to a target class. This paradigm has proven its effectiveness in the few-shot setting, even for relatively smaller models, such as BERT (Devlin et al., 2019) and RoBERTA (Liu et al., 2019), when combined with ensembling and fine-tuning (Schick and Schütze, 2021a). From the practical point of view, prompt-based learning is particularly well-suited for massive models, such as GPT-3, since it does not involve parameter tuning.

Prompt-based techniques have shown impressive performance in the few-shot setting, especially when compared to standard fine-tuning on datasets of hundreds of data points (Le Scao and Rush, 2021). However, surprisingly, the Word-in-Context task (Pilehvar and Camacho-Collados, 2019) –one of the tasks in the SuperGLUE benchmark (Wang et al., 2019)– is one exception on which these methods fail to stay on par with their fine-tuned counterparts.[‡] While a simple fine-tuned BERT-base model achieves around 69% accuracy on this task (Wang et al., 2019), GPT-3, with more than 100 times the number of parameters, performs no better than a random baseline by employing a prompt-based approach (Brown et al., 2020). The same pattern of failure is also observed in the more recent prompt based attempts (Liu et al., 2021; Schick and Schütze, 2021a).

The natural question that arises here is if the failure of few-shot techniques on WiC is due to lack of relevant encoded knowledge in PLMs or the inefficiency of the employed prompt-based methods. Two issues could be responsible for the latter case: (1) improper prompt, or (2) inefficient utilization of PLM's response. To address the first issue, there have been proposals to automatically find a suitable prompt template using a search in the discrete token space (Shin et al., 2020) or in the continuous embedding space (Liu et al., 2021). However, none of these have shown success on the WiC task.

In this work we investigate the latter issue by

[*]Work done as a Master's student at IUST.

[†]The code is freely available at https://github.com/tabasy/similarity_prompting

[‡]Given an ambiguous target word in two different contexts, the task in WiC is defined as a simple binary classification problem to identify if the triggered meaning of the target word differs in the two contexts or not.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 325 - 332

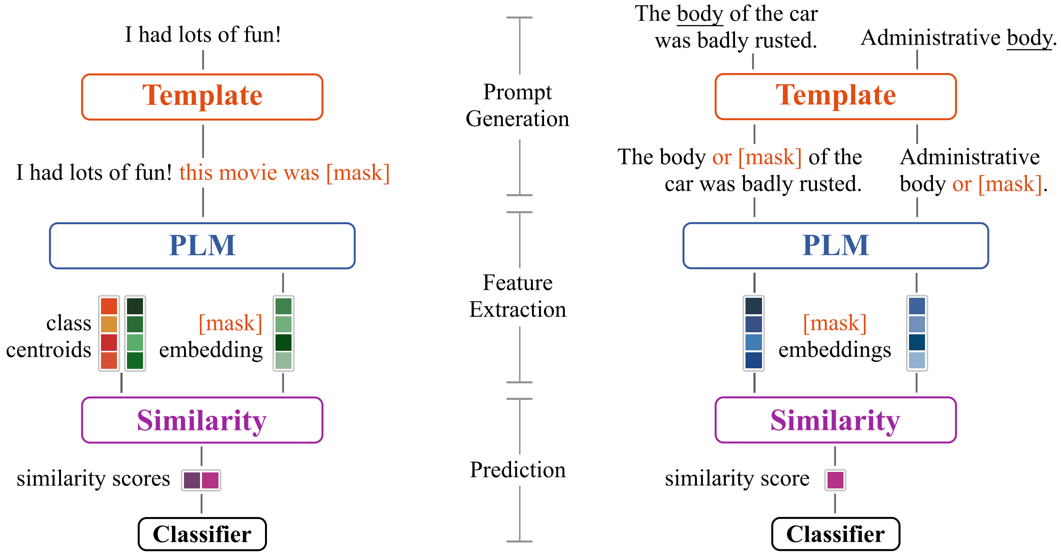


Figure 1: An illustration of the similarity-based method applied to sentiment analysis (left) and WiC (right).

introducing a new configuration for prompting. Given the comparison-based nature of WiC, we hypothesize that conventional prompting methods fall short since they only utilize a single prompt response. Hence, instead of relying on a single response, we make use of the similarity of PLM's response to the combination of a pair of prompts. The experimental results on the WiC dataset shows that, with only 16 instances per class, our proposed prompt-based technique can achieve comparable results to the fine-tuned models (with access to full training data of 2700+ instances per class). Moreover, we show that with few adjustments, this simple approach can be effectively used for other downstream tasks.

2 Methodology

Fine-tuning on a specific task can potentially update PLMs on what the task is and how to solve it. Assuming that PLMs know how to solve some tasks (to some extent), prompt-based learning focuses on the former, i.e., teaching the model what the task is, without needing to resort to large amounts of data or additional parameters. The common approach in prompt-based learning is to reformulate the task as a cloze-style question. For instance, to ask about the sentiment of a movie review, one can augment the review with a cloze question like "this movie was ——". Existing methods often pick a set of one or few word predictions as a representative for each class, utilizing the language model's response in a sub-optimal manner. We propose a similarity-based method that not only better exploits the response, but also allows using multiple prompts which paves the way for comparison-based tasks, such as WiC. In what follows in this section, we describe our similarity-based prompting approach which we will refer to as **SP** (Similarity Prompting).

As shown in Figure 1, SP consists of three main steps: (1) prompt generation, (2) feature extraction, and (3) prediction. Given a task-specific input consisting of one or more text sequences, we first use a template function to generate a prompt—a sequence of tokens containing one [MASK] token—per input sequence. For instance, in sentiment analysis, for the movie review "Just give it a chance.", a valid template function would generate as output prompt: "Just give it a chance. this movie was ——.". The next step is feature extraction from a PLM. This is done by giving the generated prompts to the PLM as input and obtaining its contextualized embedding at the MASK index.

The third step is where SP differs from existing prompt-based approaches. Here, we first obtain class-specific centroids by taking the average of the MASK embeddings of our few training examples. To classify a new sample at inference time, a simple approach would be to employ a nearest centroid classifier. However, this assumes the variance of different classes to be equal in the embedding space. To alleviate the problem, we perform a class centroid-based dimension reduction (i.e. by taking the similarity to each centroid as a feature), and train a simple linear classifier. This linear model is then used at inference time to evaluate SP on test set.

2.1 Similarity Prompting for WiC

The surprising failure of existing prompt-based techniques on the Word-in-Context task (Pilehvar and Camacho-Collados, 2019, WiC), motivated us to focus on filling this gap. Given an ambiguous target word in two different contexts, the task in WiC is defined as a simple binary classification problem to identify if the triggered meaning of the target word differs in the two contexts or not.

Previous work has fallen short of designing a single prompt template which make the PLM answer about the target word having the same meaning or not (e.g., with "yes" or "no"). Therefore, we ask PLM about the triggered meaning of the target word, separately for each context, and leave the comparison to similarity measures. Having an input sentence and the target word index, we insert “or ——” after the target word, where “——” indicates the MASK token. In the first step of SP, we apply this template function to both input sentences which generates a pair of prompts. Next the prompts are separately fed to PLM, resulting in a pair of mask embeddings as PLM’s response. Finally, our classification step reduces to that of directly comparing our pair of embedding vectors using a similarity function, to produce a single similarity score for each instance. We then train the same linear model as before on the similarity scores of the training set examples to find the best discriminating threshold.

Similarity Measures. We opted for two similarity metrics: cosine similarity and Spearman’s rank correlation. The latter is a rank-based comparison measure which is insensitive to the absolute values of individual dimensions (rather checks for their relative rankings).

3 Experiments

3.1 Comparison Systems

We compare our results on WiC with three other methods, all of which use 32 examples for their training. PET (Schick and Schütze, 2021b) prefers ALBERT-xxlarge-v2 (Lan et al., 2019) over RoBERTa (with an average gain of 8 points on a subset of SuperGLUE tasks) and fine-tunes it with manually engineered cloze-style prompts. P-tuning (Liu et al., 2021) uses the same PLM as PET, but optimizes a continuous prompt instead of tuning PLM parameters. GPT3 (Brown et al., 2020) is different in that it employs the so-called in-context learning which involves no parameter tuning.

3.2 Tasks

In addition to WiC, we also carried out experiments on two more tasks. The goal of this additional experiment is twofold: first, to show the applicability of SP to other settings, including tasks with single input sequence; and second, to evaluate if SP is effective when using prompt templates from other techniques, including those optimized for specific tasks. For this experiment, we compare against AutoPrompt (Shin et al., 2020). The approach makes use of full training set to optimize discrete prompts for each specific target task. Following AutoPrompt, we report results for the following two task:

SST. Stanford Sentiment Treebank (Socher et al., 2013) contains fine-grained sentiment labeled parse trees of sentences from movie reviews. Systems are evaluated either on a five-way fine-grained or binary classification task. We follow the latter (SST-2) in our experiments. For this task we used the automatically-generated template of AutoPrompt, along with the following manual template: $T(sent) = sent$ + “ this movie was ——.”, where $sent$ is the input sentence and “+” is concatenation operator. This is the same manual prompt used in AutoPrompt.

SICK. Sentences Involving Compositional Knowledge (Marelli et al., 2014) is a collection of sentence pairs annotated with their entailment relationship as well as a quantified measurement of their semantic similarity. In our experiments, we only use the former annotations (SICK-E) to compare our results with AutoPrompt, which only reports results for its optimized prompt. Thus we define our own manual template function as: $T(pre, hyp) = pre$ + “? Answer: ——, ” + hyp, where pre is the premise and hyp is the hypothesis of an input example.

3.3 Setup

To train our models, we only used 16 examples per class. As for PLM, we opted for RoBERTa-large to be able to benchmark our results against AutoPrompt’s (Shin et al., 2020). Our experiments are

Method	WiC	
	dev	test
Random Baseline	50.0	50.0
Fine-tuned RoBERTa-Large	-	69.9
GPT3 few-shot (Brown et al., 2020)	55.3	49.4
PET (Schick and Schütze, 2021b)	52.4	50.7
P-tuning (Liu et al., 2021)	56.3	-
Similarity Prompting - Cosine	60.3±0.4	63.6±0.5
Similarity Prompting - Spearman	**69.4±1.4**	**70.2±1.3**

Table 1: Accuracy percentage scores for Word-in-Context task. SP models are based on RoBERTa-Large.

Method	SST-2	SICK-E	
		Standard	Balanced
Majority baseline	50.0	56.7	33.3
Fine-tuned BERT	93.5	86.7	84.0
Manual Prompt			
AutoPrompt	85.2	-	-
SP-Cosine	89.1±2.1	77.3±1.5	79.8±0.8
SP-Spearman	89.2±1.8	76.6±2.3	79.0±1.0
Auto-generated Prompt			
AutoPrompt	91.4	65.0	69.3
SP-Cosine	90.7±2.3	62.1±1.0	63.2±1.9
SP-Spearman	91.8±1.5	61.6±0.7	62.2±1.6

Table 2: Test set accuracy on SST-2 and SICK-E tasks. SP and AutoPrompt (Shin et al., 2020) methods are based on RoBERTa-Large.

repeated 5 times using different randomly sampled training examples. For each experiment, we report the average performance along with the standard deviation.

3.4 Results

Given that our experiments are mainly focused on the WiC dataset, we first report our results on this benchmark, and then provide additional results for the other two tasks.

3.4.1 WiC

Table 1 summarizes the results on WiC with RoBERTa-Large as SP's PLM. The performance of SP in the few-shot setting is in the same ballpark as supervised fine-tuning (with nearly 170 times the data, i.e., 2,714 instances per class). This observation suggests that PLMs already encode a certain amount of task-related knowledge and the supervised fine-tuning mainly updates their task description (i.e., what the task is, not how to solve it). Therefore, using limited examples in the few-shot setting they are able to reach their maximum fine-tuning potential on WiC. We report SP's performance on WiC for other PLMs in the Appendix which shows our method/observation does not depend on a specific PLM. We also include some detailed examples of how SP works for WiC in the Appendix.

3.4.2 SICK and SST-2

The results on SST-2 and SICK-E are shown in Table 2. We compare SP with AutoPrompt which searches for the best template for each task. For SST-2, we observe that SP can exploit a manual prompt template significantly better than AutoPrompt, while being competitive using the best template optimized by AutoPrompt (auto-generated). This suggests that it is possible to gain significant improvement by simply exploiting a non-optimized manual prompt template.

To compare our results with AutoPrompt on the SICK-E task, we report accuracy score of SP for the standard test set (with neutral majority) and its balanced variant. SP retains an acceptable level of performance, particularly with the manual prompt, but lags behind with the auto-generated prompt. We note that the goal of this experiment was to showcase that our simple adaptation is also applicable to scenarios other than the setting of WiC. In fact, one could argue that the auto-generated prompt of AutoPrompt is sub-optimal for our model, which results in dropped performance on the SICK-E dataset.

3.5 Similarity Measures Comparison

Notably, the Spearman correlation score, which is less commonly used for comparing embeddings, outperforms the cosine similarity on WiC by a large margin while maintaining the same level of performance on other tasks. This superiority can be explained by the assumption that cosine similarity is more susceptible to variations in the dominant dimensions. To evaluate this hypothesis, we performed an experiment in which the most dominant dimension was set to zero for all the embeddings (the dominant dimension is identical across all vectors). The results approve the assumption: pruned cosine similarity gains around 10% absolute performance boost on WiC, filling the gap to Spearman correlation. However, the gain in the other two tasks is negligible.

The difference in the gain across tasks can be explained by the difference in their underlying nature.

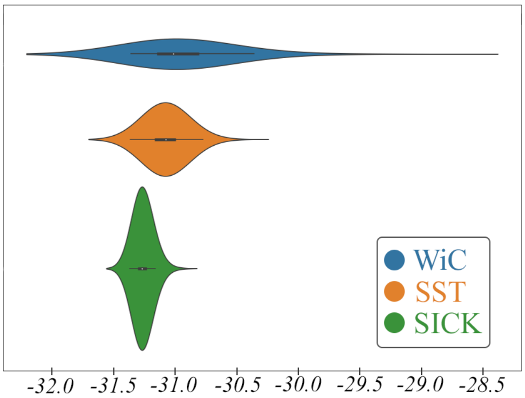


Figure 2: The distribution of values for the most dominant dimension of the MASK embedding for 1200 samples for the three tasks.

In WiC, the MASK embeddings can potentially refer to any word, varying from sample to sample. However, in SST and SICK the MASK template embedding is more restricted, often representing a closely related word to one of the class centroid embeddings (e.g., in SST the MASK embedding almost always represents a positive or negative adjective). This results in a higher spread on the most dominant dimension in the case of WiC. It is known that the most dominant dimensions in PLMs often encode irrelevant information, such as word frequency (Gao et al., 2019), therefore hampering performance for sensitive metrics such as cosine similarity. To verify our hypothesis, we ran an experiment using 1200 sample MASK embeddings for each of our three tasks. Figure 2 illustrates the distribution of values for the most dominant dimension. The ratio of variance is 6.5 times for WiC compared to SST and 27.3 times compared to SICK. This further supports the sensitivity of cosine similarity for WiC to the noisy variations along the most dominant dimension compared to the other two tasks.

4 Conclusion

We proposed an adaptation of prompt-based learning which addresses the common failure of existing techniques on the WiC dataset. In this work we showed that similarity based approach to prompt-based learning is capable of achieving comparable results to purely fine-tuning based methods on Word-in-Context task, in which previous few-shot attempts have failed. We also showed that Spearman's ranking correlation is a more robust choice of similarity measure compared to cosine similarity in this setting. We hope that our positive results inspire other prompting strategies to better exploit the encoded knowledge in PLMs. As future work, one interesting direction could be to perform further analysis on the behaviour of Spearman's correlation compared to cosine similarity anywhere it is applicable as a similarity measure.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

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

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tieyan Liu. 2019. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Teven Le Scao and Alexander Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, Online. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach.

M. Marelli, S. Menini, Marco Baroni, L. Bentivogli, R. Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *LREC*.

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

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Wilson L. Taylor. 1953. "Cloze Procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. *Advances in Neural Information Processing Systems*, 32.

A Experiments with other PLMs

This appendix contains more details on WiC experiments. Table 3 shows full test set results of SP for different PLMs and similarity measures to compare the performance of SP in different scenarios. Since our cloze-style prompt template is not applicable to GPT2, we use a different template for it: $sentence$ + $targetword$ + " means ——". The results in Table 3 generally confirm the effectiveness of SP with different PLMs. Notably, this observation is in line with our previous experiments that in general Spearman has superior performance over Cosine similarity.

Base model	**Cosine**	**Spearman**
RoBERTa-Large	63.6	70.2
BERT-Large-Cased	69.4	69.0
RoBERTa-Base	63.8	68.7
BERT-Base-Cased	64.8	67.1
GPT2-Large	56.4	63.3
GPT2-Base	62.3	62.6

Table 3: Test set accuracy of SP on WiC task, based on different PLMs (both Masked language model and Causal language models) and similarity metrics.

B Qualitative Analysis

We include some examples of how SP works on WiC in Table 4 for qualitative analysis. The examples are those from WiC dev set which had negative labels. We did not include the positive examples, since the observation that the same words with the same senses are treated similarly, might not provide a useful insight. The table presents our generated prompts, top-5 most probable words predicted by RoBERTa-Large for each prompt and the final prediction of SP. The top three examples are correctly predicted as negative with high confidence (high similarity score), while the bottom three are predicted positive again with high confidence. The most probable predicted words for the top three examples indicate that the PLM has spotted the correct senses in both contexts. For the bottom three where the model fails, we can observe that the target words have very similar or close senses, making them really hard to distinguish.

Prompt1 (Top-5 words)	**Prompt2 (Top-5 words)**	**Prediction**	**Ground Truth**
The drawing or —— of water from the well. (use, extraction, taking, pumping, consumption)	He did complicated pen-and-ink drawings or —— like medieval miniatures. (paintings, sculptures, something, more, looked)	Not matched	Not matched
The body or —— of the car was badly rusted. (trunk, roof, chassis, frame, grill)	Administrative body or ——. (agency, institution, government, commission, equivalent)	Not matched	Not matched
The main body of the sound or —— ran parallel to the coast. (river, bay, sea, ocean, channel)	He strained to hear the faint sounds or ——. (voices, footsteps, whispers, conversations, cries)	Not matched	Not matched
He could not conceal his hostility or ——. (anger, disgust, irritation, contempt, frustration)	He could no longer contain his hostility or ——. (anger, rage, frustration, aggression, disgust)	Matched	Not matched
There was a blockage or —— in the sewer, so we called out the plumber. (something, leak, obstruction, defect, overflow)	We had to call a plumber to clear out the blockage or —— in the drainpipe. (debris, obstruction, water, leak, crack)	Matched	Not matched
The senator received severe criticism or —— from his opponent. (threats, ridicule, mockery, attacks, threat)	The politician received a lot of public criticism or —— for his controversial stance on the issue. (backlash, ridicule, mockery, condemnation, criticism)	Matched	Not matched

Table 4: Detailed examples of how SP works on WiC.

Hierarchical Curriculum Learning for AMR Parsing

Peiyi Wang[1]*, **Liang Chen**[1]*, **Tianyu Liu**[2], **Damai Dai**[1],
Yunbo Cao[2], **Baobao Chang**[1], **Zhifang Sui**[1]
[1] Key Laboratory of Computational Linguistics, Peking University, MOE, China
[2] Tencent Cloud Xiaowei
wangpeiyi9979@gmail.com; leo.liang.chen@outlook.com
{rogertyliu, yunbocao}@tencent.com
{daidamai, chbb, szf}@pku.edu.cn

Abstract

Abstract Meaning Representation (AMR) parsing aims to translate sentences to semantic representation with a hierarchical structure, and is recently empowered by pretrained sequence-to-sequence models. However, there exists a gap between their flat training objective (i.e., equally treats all output tokens) and the hierarchical AMR structure, which limits the model generalization. To bridge this gap, we propose a Hierarchical Curriculum Learning (HCL) framework with Structure-level (SC) and Instance-level Curricula (IC). SC switches progressively from core to detail AMR semantic elements while IC transits from structure-simple to -complex AMR instances during training. Through these two warming-up processes, HCL reduces the difficulty of learning complex structures, thus the flat model can better adapt to the AMR hierarchy. Extensive experiments on AMR2.0, AMR3.0, structure-complex and out-of-distribution situations verify the effectiveness of HCL.

1 Introduction

Abstract Meaning Representation (AMR) (Banarescu et al., 2013) parsing aims to translate a natural sentence into a directed acyclic graph. Figure 1(a) illustrates an AMR graph where nodes represent concepts, e.g., 'die-01' and 'soldier', and edges represent relations, e.g., ':ARG1' and ':quant'. AMR has been exploited in the downstream NLP tasks, including information extraction (Rao et al., 2017; Wang et al., 2017; Zhang and Ji, 2021), text summarization (Liao et al., 2018; Hardy and Vlachos, 2018) and question answering (Mitra and Baral, 2016; Sachan and Xing, 2016).

The powerful pretrained encoder-decoder models, e.g., BART (Lewis et al., 2020), have been successfully adapted to the AMR parsing and became the mainstream and state-of-the-art methods (Bevilacqua et al., 2021). Through directly generating the linearized AMR graph (e.g., Figure 1(a)) from the sentence, these sequence-to-sequence methods (Xu et al., 2020b; Bevilacqua et al., 2021) circumvent the complex data processing pipeline and can be easily optimized compared with transition-based or graph-based methods (Naseem et al., 2019; Lee et al., 2020; Lyu and Titov, 2018; Zhang et al., 2019a,b; Cai and Lam, 2020; Zhou et al., 2021b). However, there exists a gap between the flat sentence-to-AMR training objective[1] and AMR graphs, since sequence-to-sequence models deviate from the essence of graph representation. Therefore, it is difficult for sequential generators to learn the inherent hierarchical structure of AMR (Zhou et al., 2021b).

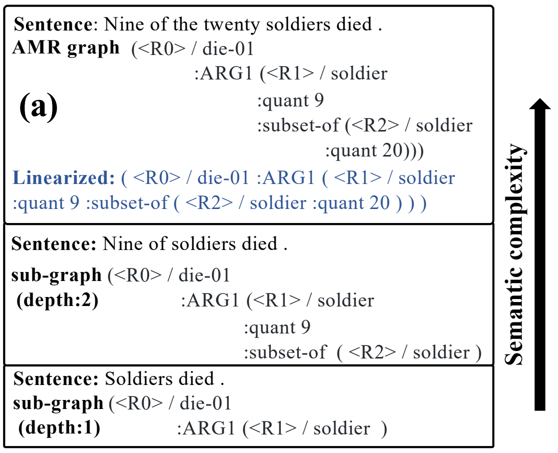

Sentence: Nine of the twenty soldiers died .
AMR graph (<R0> / die-01
:ARG1 (<R1> / soldier
:quant 9
:subset-of (<R2> / soldier
:quant 20)))
(a)
Linearized: (<R0> / die-01 :ARG1 (<R1> / soldier :quant 9 :subset-of (<R2> / soldier :quant 20)))
Sentence: Nine of soldiers died .
sub-graph (<R0> / die-01
(depth:2) :ARG1 (<R1> / soldier
:quant 9
:subset-of (<R2> / soldier)
Sentence: Soldiers died .
sub-graph (<R0> / die-01
(depth:1) :ARG1 (<R1> / soldier)
Semantic complexity

Figure 1: The AMR (sub-)graphs of the sentence "Nine of the twenty soldiers died". The deeper sub-graphs contain more sophisticated semantics compared with shallower ones.

Humans usually adapt to difficult tasks by dealing with examples gradually from easy to hard, i.e., Curriculum Learning (Bengio et al., 2009; Platanios et al., 2019; Su et al., 2021; Xu et al., 2020a). Inspired by human behavior, we propose a hierarchi-

*Equal Contribution.

[1] Flat means the objective equally treats all output tokens.

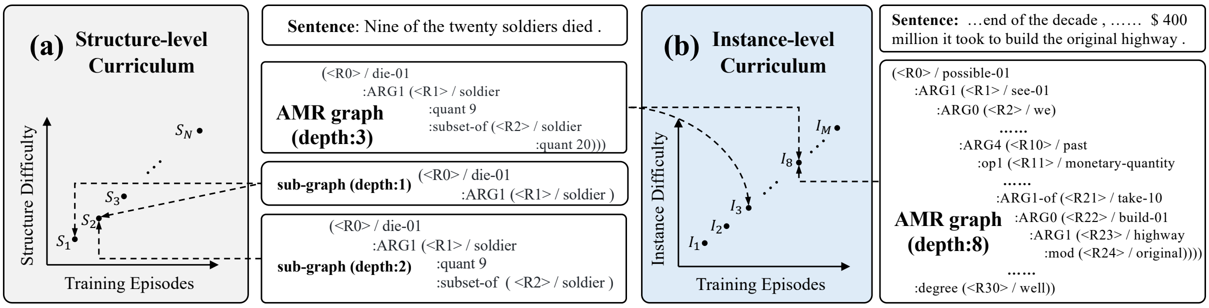


Figure 2: The overview of our hierarchical curriculum learning framework with two curricula, Structure-level (SC) and Instance-level Curricula (IC). During training, SC follows the principle of *learning core semantics first*, which switches progressively from shallow to deep AMR sub-graphs. IC follows the human intuition to *start with easy instances*, which transits from easy to hard AMR instances.

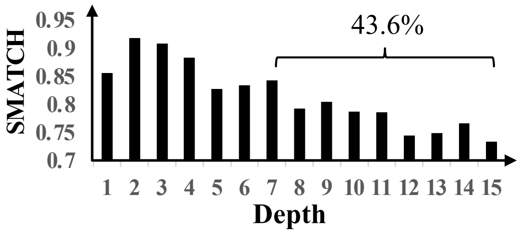


Figure 3: The average SMATCH scores for AMR graphs with different depths. The AMR graphs with at least depth 7 accounted for 43.6% in the AMR-2.0 test set.

cal curriculum learning framework with two curricular strategies to help the flat pretrained model progressively adapt to the hierarchical AMR graph. **(1) Structure-level Curriculum (SC)**. AMR graphs are organized in a hierarchy where the core semantic elements stay closely to the root node (Cai and Lam, 2019). As depicted in Figure 1, the concepts and relations that locate in the different layers of the AMR graph correspond to different levels of abstraction in terms of the semantic representation. Motivated by the human learning process, i.e., ***core concepts first, then details***, SC enumerates all AMR sub-graphs with different depths, and deals with them in order from shallow to deep. **(2) Instance-level Curriculum (IC)**. Our preliminary study in Figure 3 shows that the performance of the vanilla BART baseline would drop rapidly as the depth of AMR graph grows, which indicates that handing deeper AMR hierarchy is more difficult for pretrained models. Inspired by the human cognition, i.e., ***easy ones first, then hard ones***, we propose IC which trains the model by starting from easy instances with a shallower AMR structure and then handling hard instances.

To sum up: (1) Inspired by the human learning process, i.e., *core concepts first* and *easy instances first*, we propose a hierarchical curriculum learning (HCL) framework to help the sequence-to-sequence model progressively adapt to the AMR hierarchy. (2) Extensive experiments on AMR2.0, AMR3.0, structure-complex and out-of-distribution situations verify the effectiveness of HCL.

2 Methodology

We formulate AMR parsing as a sequence-to-sequence transformation. Given a sentence $\mathbf{x} = (x_1, ..., x_N)$, the model aims to generate a linearized AMR graph $\mathbf{y} = (y_1, ..., y_M)$. As shown in Figure 1(a), following Bevilacqua et al. (2021), the AMR graph is linearized by the DFS-based linearization method with special tokens to indicate variables and parentheses to mark visit depth. Specifically, variables of AMR nodes are set to a series of special tokens *<R0>, ..., <Rk>* (more details of linearization are included in Appendix A). In this paper, we propose a hierarchical curriculum learning framework (Figure 2) with the structure- and instance-level curricula to help the flat model progressively adapt to the structured AMR graph.

2.1 Structure-level Curriculum

Motivated by *learning core concepts first*, we propose Structure-level Curriculum (SC). AMR graphs are organized in a hierarchy where the core semantics stay closely to the root (Cai and Lam, 2019), thus SC divides all AMR sub-graphs into N buckets according to their depths $\{S_i : i = 1, 2, ..., N\}$, where S_i contains AMR sub-graphs with the depth i. As shown in Figure 2(a), SC has N training episodes, and each episode consists of T_{sc} steps. In each step of the i-th episode, the training scheduler samples a batch of examples from buckets $\{S_j : j \leq i\}$ to train the model. When parsing

	Model	SMATCH	Structure-independent					Structure-dependent		
			NoWSD	Conc.	NER	Neg.	Wiki.	Unll.	Reen.	SRL
AMR2.0	Lyu and Titov (2018)^G	74.4	75.5	85.9	86.0	58.4	75.7	77.1	52.3	69.8
	Zhang et al. (2019a)^G	76.3	76.8	84.8	77.9	75.2	85.8	79.0	60.0	69.7
	Cai and Lam (2020)	78.7	79.2	88.1	87.1	66.1	81.3	81.5	63.8	74.5
	Cai and Lam (2020)^G	80.2	80.8	88.1	81.1	**78.9**	**86.3**	82.8	64.6	74.2
	Fernandez Astudillo et al. (2020)	80.2	80.7	88.1	87.5	64.5	78.8	84.2	70.3	78.2
	Zhou et al. (2021a)	81.7	82.3	88.7	88.5	69.7	78.8	85.5	71.1	80.8
	Bevilacqua et al. (2021)	83.8	84.4	**90.2**	90.6	74.4	84.3	86.1	70.8	79.6
	HCL (Ours)	**84.3**	**85.0**	**90.2**	**91.6**	75.9	84.0	**87.7**	**74.5**	**83.2**
AMR3.0	Cai and Lam (2020)	78.0	78.5	88.5	83.7	68.9	75.7	81.9	63.7	73.2
	Cai and Lam (2020)^G	76.7	77.2	86.5	74.7	72.6	77.3	80.6	62.6	72.2
	Zhou et al. (2021a)	80.3	-	-	-	-	-	-	-	-
	Bevilacqua et al. (2021)	83.0	83.5	**89.8**	87.2	**73.0**	**82.7**	85.4	70.4	78.9
	HCL (Ours)	**83.7**	**84.2**	89.5	**89.0**	**73.0**	82.6	**86.9**	**73.9**	**82.4**

Table 1: SMATCH and fine-grained F1 scores on the AMR 2.0 and 3.0 test set. Our results are the average of 3 runs with different random seeds. Models^G indicate models with graph re-categorization (a data processing method that may hurt the model generalization ability Bevilacqua et al. (2021)).

a sentence into a sub-graph with the depth d, we append a special string "parse to d layers" to the input sentence, and replace the start token of the decoder with an artificial token <d>, so the model can perceive layers that need to be parsed.

2.2 Instance-level Curriculum

Inspired by *learning easy instances first*, we propose Instance-level Curriculum (IC). Figure 3 shows AMR graphs with deeper layers can be regarded as harder instances for the flat pretrained model, thus IC divides all AMR graphs into M buckets according to their depths $\{I_i : i = 1, ..., M\}$, where I_i contains AMR graphs with the depth i. As shown in Figure 2(b), IC has M training episodes, and each episode consists of T_{ic} steps. In each step of the i-th episode, the training scheduler samples a batch of examples from buckets $\{I_j : j \leq i\}$ to train the model. Specifically, we first use SC and then IC to train the model, since SC (follows learning core semantics first) is for AMR sub-graphs, which can be regarded as a warming-up stage of IC (obeys learning easy instances first), which is for AMR full graphs.

3 Experiments

Datasets and Evaluation Metrics We evaluate our hierarchical curriculum learning framework on two popular AMR benchmarks, AMR2.0 (LDC2017T10) and AMR3.0 (LDC2020T02). Please refer to the Appendix B for details of two benchmarks. Following Bevilacqua et al. (2021), we use the SMATCH scores (Cai and Knight, 2013) and the fine-grained evaluation metrics (Damonte et al., 2017)[2] to evaluate the performances.

Experiment Setups Our implementation is based on Huggingface's transformers library (Wolf et al., 2020) and the open codebase of Bevilacqua et al. (2021)[3]. We use BART-large as our sequence-to-sequence model the same as Bevilacqua et al. (2021). We utilizes RAdam (Liu et al., 2020) as our optimizer with the learning rate 3e-5. The batch size is 2048 graph linearization tokens with the gradient accumulation 10. Dropout is set to 0.25 and beam size is 5. The training steps T_{sc} is 1000 and T_{ic} is 500. After the curriculum training, the model is trained for 30 epochs on the training set. We use cross-entropy as our loss function. We train our model on a single NVIDIA TESLA V100 GPU with 32GB memory. We adopt the same post-processing process as Bevilacqua et al. (2021). Our code and model are available at https://github.com/Wangpeiyi9979/HCL-Text2AMR.

Main Results We compare our method with previous approaches in Table 1. As is shown, on AMR2.0 and AMR3.0, our hierarchical curriculum learning model achieves 84.3 ± 0.1 and 83.7 ± 0.1 SMATCH scores, and outperforms Bevilacqua et al. (2021) 0.5 and 0.7 SMATCH scores, respectively. For the fine-grained results, our model achieves the best performance in 6 out of 8 metrics on both AMR2.0 and AMR3.0, which shows the effective-

[2] https://github.com/mdtux89/amr-evaluation

[3] https://github.com/SapienzaNLP/spring

Model	AMR2.0	AMR3.0
Ours	84.3	83.7
w/o instance curriculum	84.1	83.5
w/o structure curriculum	84.0	83.3
w/o curricula	83.8	83.0

Table 2: The effect of our proposed curricula on the test set of AMR2.0 and AMR3.0. 'w/o' denotes without.

ness of our method. Although Cai and Lam (2020) outperforms our model in Neg. and Wiki. on AMR2.0, they adopt a complex process, which may hurt the model generalization ability. Bevilacqua et al. (2021) outperforms slightly our model in Conc. and Wiki. on AMR3.0. However, these metrics are unrelated to the AMR structure that our HCL focuses on.

4 Analysis

Structure Benefit In order to explore the effectiveness of our HCL framework for the structured AMR parsing. We divide the fine-grained F1 scores into 2 categories, "structure-dependent" (unlabelled, re-entrancy and SRL) and "structure-independen" (the left 5 metrics). Please refer to Appendix C for the reason for this division. As shown in Table 1, compared with Bevilacqua et al. (2021) (also a sequence-to-sequence model based on BART-large), our method achieves 2.97 and 2.83 average F1 scores improvement on 3 structure-dependent metrics on AMR2.0 and AMR3.0, respectively, which proves HCL helps the flat sequence-to-sequence model better adapt to AMR with the hierarchical and complex structure.

Hard Instances Benefit Figure 4 shows the performances of our HCL and Bevilacqua et al. (2021) (SPRING) at different layers. As is shown, as the number of layers increases, HCL exceeds SPRING greater, which shows our HCL helps the model better handle hard instances.[4] In addition, to some extend, out-of-distribution (OOD) instances can be regarded as hard instances, thus we also consider the OOD situation. Bevilacqua et al. (2021) propose the OOD evaluation for AMR parsers. Following Bevilacqua et al. (2021), we train our model on the training dataset of AMR2.0, and then evaluate it on 3 OOD test datasets, BIO, TLP and News3. Please refer to Appendix B for details of OOD datasets. As shown in Table 3, our method outperforms Bevilacqua et al. (2021) on all 3 OOD datasets, which shows our HCL framework can also improve the generalization ability of the model.

Model	BIO	TLP	News3
Bevilacqua et al. (2021)	59.7	77.3	73.7
HCL (Ours)	**61.1**	**78.2**	**75.3**

Table 3: Results on out-of-distribution data.

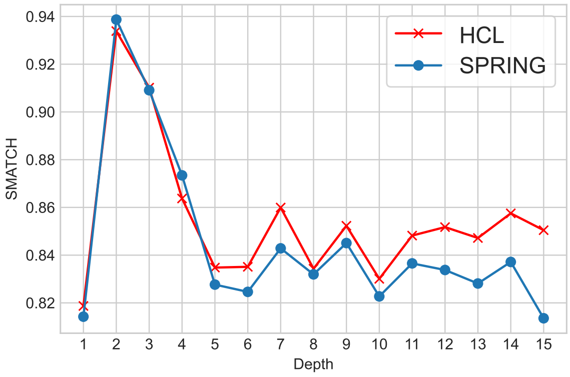


Figure 4: The average SMATCH scores of our HCL and Bevilacqua et al. (2021) (SPRING) at different depths on the AMR2.0 test set.

Ablation Study To illustrate the effect of our proposed curricula. We conduct ablation studies by removing one curriculum at a time. Table 2 shows the SMATCH scores on both AMR2.0 and AMR3.0. As shown in Table 2, we can see both curricula are conducive to the performance of the model, and they are complementary to each other. Specifically, the structure-level curriculum (SC) is more effective than the instance-level curriculum (IC). We think the reason is that SC constructs AMR subgraphs for training, which enhances the model's ability to perceive the AMR hierarchy.

5 Conclusion

In this paper, we propose a Hierarchical Curriculum Learning (HCL) framework for sequence-to-sequence AMR parsing, which consists of Structure-level Curriculum (SC) and Instance-level Curriculum (IC). inspired by human cognition, SC follows the principle of learning the core concepts of AMR first, and IC obeys the rule of learning easy instances first. SC and IC train the model on different hierarchies (AMR sub-graphs and AMR full graphs). Extensive experiments on AMR2.0, AMR3.0, structure-complex and out-of-distribution situations verify the effectiveness of HCL.

[4] An intuitive case study for the hard instance parsing is included in Appendix D.

Acknowledgement

The authors would like to thank the anonymous reviewers for their thoughtful and constructive comments, and Bevilacqua et al. (2021) for their high-quality open codebase. This paper is supported by the National Key Research and Development Program of China under Grant No. 2020AAA0106700, the National Science Foundation of China under Grant No.61936012 and 61876004, and NSFC project U19A2065.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML 2009, page 41–48, New York, NY, USA. Association for Computing Machinery.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*.

Deng Cai and Wai Lam. 2019. Core semantic first: A top-down approach for AMR parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3799–3809, Hong Kong, China. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2020. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2017. An incremental parser for abstract meaning representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 536–546, Valencia, Spain. Association for Computational Linguistics.

Ramón Fernandez Astudillo, Miguel Ballesteros, Tahira Naseem, Austin Blodgett, and Radu Florian. 2020. Transition-based parsing with stack-transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1001–1007, Online. Association for Computational Linguistics.

Hardy Hardy and Andreas Vlachos. 2018. Guided neural language generation for abstractive summarization using Abstract Meaning Representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 768–773, Brussels, Belgium. Association for Computational Linguistics.

Young-Suk Lee, Ramón Fernandez Astudillo, Tahira Naseem, Revanth Gangi Reddy, Radu Florian, and Salim Roukos. 2020. Pushing the limits of AMR parsing with self-learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3208–3214, Online. Association for Computational Linguistics.

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

Kexin Liao, Logan Lebanoff, and Fei Liu. 2018. Abstract meaning representation for multi-document summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1178–1190.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the variance of the adaptive learning rate and beyond. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Chunchuan Lyu and Ivan Titov. 2018. AMR parsing as graph prediction with latent alignment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 397–407, Melbourne, Australia. Association for Computational Linguistics.

Arindam Mitra and Chitta Baral. 2016. Addressing a question answering challenge by combining statistical methods with inductive rule learning and reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Tahira Naseem, Abhishek Shah, Hui Wan, Radu Florian, Salim Roukos, and Miguel Ballesteros. 2019. Rewarding Smatch: Transition-based AMR parsing with reinforcement learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4586–4592, Florence, Italy. Association for Computational Linguistics.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172, Minneapolis, Minnesota. Association for Computational Linguistics.

Sudha Rao, Daniel Marcu, Kevin Knight, and Hal Daumé III. 2017. Biomedical event extraction using Abstract Meaning Representation. In *BioNLP 2017*, pages 126–135, Vancouver, Canada,. Association for Computational Linguistics.

Mrinmaya Sachan and Eric Xing. 2016. Machine comprehension using rich semantic representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Berlin, Germany. Association for Computational Linguistics.

Yixuan Su, Deng Cai, Qingyu Zhou, Zibo Lin, Simon Baker, Yunbo Cao, Shuming Shi, Nigel Collier, and Yan Wang. 2021. Dialogue response selection with hierarchical curriculum learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1740–1751, Online. Association for Computational Linguistics.

Yanshan Wang, Sijia Liu, Majid Rastegar-Mojarad, Liwei Wang, Feichen Shen, Fei Liu, and Hongfang Liu. 2017. Dependency and amr embeddings for drug-drug interaction extraction from biomedical literature. In *Proceedings of the 8th acm international conference on bioinformatics, computational biology, and health informatics*, pages 36–43.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. 2020a. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, Online. Association for Computational Linguistics.

Dongqin Xu, Junhui Li, Muhua Zhu, Min Zhang, and Guodong Zhou. 2020b. Improving AMR parsing with sequence-to-sequence pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2501–2511, Online. Association for Computational Linguistics.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019a. Amr parsing as sequence-to-graph transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 80–94, Florence, Italy. Association for Computational Linguistics.

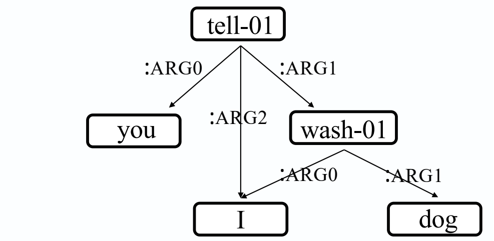
Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019b. Broad-coverage semantic parsing as transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3786–3798, Hong Kong, China. Association for Computational Linguistics.

Zixuan Zhang and Heng Ji. 2021. Abstract Meaning Representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–49, Online. Association for Computational Linguistics.

Jiawei Zhou, Tahira Naseem, Ramón Fernandez Astudillo, and Radu Florian. 2021a. AMR parsing with action-pointer transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5585–5598, Online. Association for Computational Linguistics.

Jiawei Zhou, Tahira Naseem, Ramón Fernandez Astudillo, Young-Suk Lee, Radu Florian, and Salim Roukos. 2021b. Structure-aware fine-tuning of sequence-to-sequence transformers for transition-based AMR parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6279–6290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Linearzation



DFS Linearization:

(<R0> tell-01 :ARG0 (<R1> you) :ARG1 (<R3> wash-01 : ARG0 <R2> :ARG1 (<R4> dog)) :ARG2 (<R2> I))

Figure 5: The linearization for the AMR graph of the sentence "You told me to wash the dog".

As shown in Figure 5, following Bevilacqua et al. (2021), the AMR graph is linearized by the DFS-based linearization method according to the edge order (':ARG0'→':ARG1'→':ARG2'). Variables of the AMR graph are set to a series of special tokens *<R0>, <R1>, <R2>, <R3>, <R4>*, and the depth is marked by parentheses.

B Datasets

B.1 In-domain Distribution

AMR2.0 (LDC2017T10) contains 36,521, 1,368 and 1,371 sentence-AMR pairs in training, development and testing sets, respectively.

AMR3.0 (LDC2020T02) is larger than AMR2.0 in size, which contains 55,635, 1,722 and 1,898 sentence-AMR pairs for training development and testing set, respectively. AMR3.0 is a superset of AMR2.0.

B.2 Out-domain Distribution

BIO is a test set of the Bio-AMR corpus, consisting of 500 instances.

TLP is a AMR dataset annoated on the children's novel *The Little Prince* (version 3.0), consisting of 1,562 instances.

New3 is a sub-set of AMR3.0, which is not included in the AMR2.0 training set, consisting of 527 instances.

C Fine-grained Metric Division

There are 8 fine-grained AMR metrics: (1) **Unlabeled**: Smatch score computed on the predicted graphs after removing all edge labels. (2) **No WSD.**: Smatch score while ignoring Propbank senses (e.g., duck-01 vs duck-02). (3) **Named Ent.**: F-score on the named entity recognition (:name roles). (4) **Wikification**: F-score on the wikification (:wiki roles). (5) **Negation**: F-score on the negation detection (:polarity roles). (6) **Concepts**: F-score on the concept identification task. (7) **Reentrancy**: Smatch computed on reentrant edges only, e.g., the edges of node 'I' in Figure A. (8) **SRL**: Smatch computed on :ARG-i roles only.

We only regard Unlabeled, Reentrancy and SRL as "structure-dependent" metrics, since: (1) Unlabeled does not consider any edge labels, and only considers the graph structure. (2) Reentrancy is a typical structure feature for the AMR graph. Without reentrant edges, the AMR graph is reduced to a tree. (3) SRL denotes the core-semantic relation of the AMR, which determines the core structure of the AMR. (4) As described above, all other metrics have little relationship with the structure.

D Case Study

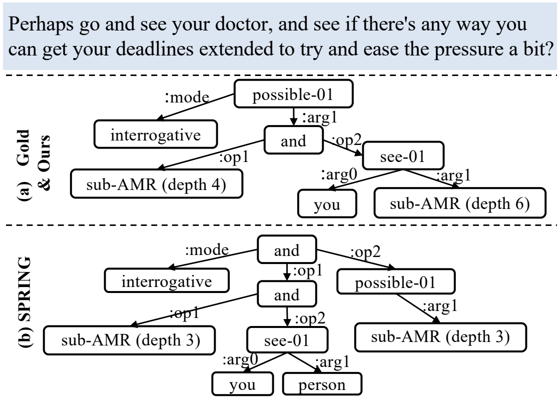

Perhaps go and see your doctor, and see if there's any way you can get your deadlines extended to try and ease the pressure a bit?

Figure 6: A specific case from the test set of AMR2.0. For the input sentence, our method achieves the right AMR, while the baseline model (i.e., SPRING (Bevilacqua et al., 2021)) gets a shallower and wrong structure AMR.

Figure 6 shows a case study (we omit some details of AMR graphs for a more clear description). As is illustrated, our method achieves the right AMR for the input sentence. However, the AMR parsed by the SPRING model (depth:5) is shallower than the gold AMR (depth:9), and their structures are also different (e.g., the root of the gold AMR and the SPRING parsed AMR are 'possible-01' and 'and', respectively). This case intuitively shows our HCL framework can help the model better handle the hard instance with complex structure.

PARE: A Simple and Strong Baseline for Monolingual and Multilingual Distantly Supervised Relation Extraction

Vipul Rathore* **Kartikeya Badola*** **Parag Singla** **Mausam**
Indian Institute of Technology
New Delhi, India
rathorevipul28@gmail.com, kartikeya.badola@gmail.com
parags@cse.iitd.ac.in, mausam@cse.iitd.ac.in

Abstract

Neural models for distantly supervised relation extraction (DS-RE) encode each sentence in an entity-pair bag separately. These are then aggregated for bag-level relation prediction. Since, at encoding time, these approaches do not allow information to flow from other sentences in the bag, we believe that they do not utilize the available bag data to the fullest. In response, we explore a simple baseline approach (*PARE*) in which all sentences of a bag are concatenated into a *passage* of sentences, and encoded jointly using BERT. The contextual embeddings of tokens are aggregated using attention with the candidate relation as query – this summary of whole passage predicts the candidate relation. We find that our simple baseline solution outperforms existing state-of-the-art DS-RE models in both monolingual and multilingual DS-RE datasets.

1 Introduction

Given some text (typically, a sentence) t mentioning an entity pair (e_1, e_2), the goal of relation extraction (RE) is to predict the relationships between e_1 and e_2 that can be inferred from t. Let $B(e_1, e_2)$ denote the set of all sentences (bag) in the corpus mentioning e_1 and e_2 and let $R(e_1, e_2)$ denote all relations from e_1 to e_2 in a KB. Distant supervision (DS) trains RE models given $B(e_1, e_2)$ and $R(e_1, e_2)$, without sentence level annotation (Mintz et al., 2009). Most DS-RE models use the "at-least one" assumption: $\forall r \in R(e_1, e_2)$, $\exists t^r \in B(e_1, e_2)$ such that t^r expresses (e_1, r, e_2).

Recent neural approaches to DS-RE encode each sentence $t \in B(e_1, e_2)$ and then aggregate sentence embeddings using an aggregation operator – the common operator being intra-bag attention (Lin et al., 2016). Various models differ in their approach to encoding (e.g., PCNNs, GCNs, BERT) and their loss functions (e.g., contrastive learning, MLM), but agree on the design choice of encoding each sentence independently of the others (Vashishth et al., 2018; Alt et al., 2019; Christou and Tsoumakas, 2021; Chen et al., 2021). We posit that this choice leads to a suboptimal usage of the available data – information from other sentences might help in better encoding a given sentence.

We explore this hypothesis by developing a simple baseline solution. We first construct a *passage* $P(e_1, e_2)$ by concatenating all sentences in $B(e_1, e_2)$. We then encode the whole passage through BERT (Devlin et al., 2019) (or mBERT for multilingual setting). This produces a contextualized embedding of every token in the bag. To make these embeddings aware of the candidate relation, we take a (trained) relation query vector, $\mathbf{r}$, to generate a relation-aware summary of the whole passage using attention. This is then used to predict whether (e_1, r, e_2) is a valid prediction.

Despite its simplicity, our baseline has some conceptual advantages. First, each token is able to exchange information with other tokens from other sentences in the bag – so the embeddings are likely more informed. Second, in principle, the model may be able to relax a part of the at-least-one assumption. For example, if no sentence individually expresses a relation, but if multiple facts in different sentences collectively predict the relation, our model may be able to learn to extract that.

We name our baseline model Passage-Attended Relation Extraction, *PARE* (*mPARE* for multilingual DS-RE). We experiment on four DS-RE datasets – three in English, NYT-10d (Riedel et al., 2010), NYT-10m, and Wiki-20m (Gao et al., 2021), and one multilingual, DiS-ReX (Bhartiya et al., 2022). We find that in all four datasets, our proposed baseline significantly outperforms existing state of the art, yielding up to 5 point AUC gain. Further attention analysis and ablations provide additional insight into model performance. We re-

[*] Equal Contribution

May 22-27, 2022 ©2022 Association for Computational Linguistics

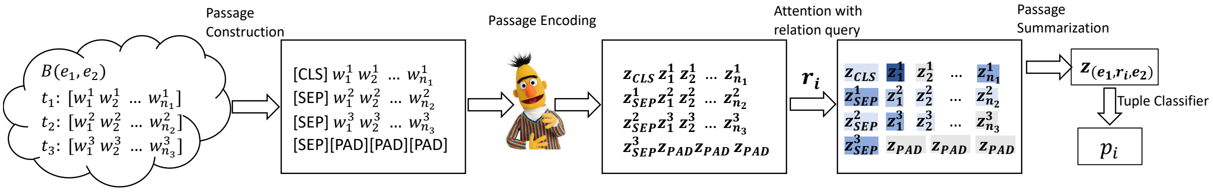

Passage Construction
Passage Encoding
Attention with relation query
Passage Summarization
$B(e_1, e_2)$
t_1: $[w_1^1\ w_2^1\ \dots\ w_{n_1}^1]$
t_2: $[w_1^2\ w_2^2\ \dots\ w_{n_2}^2]$
t_3: $[w_1^3\ w_2^3\ \dots\ w_{n_3}^3]$
[CLS] $w_1^1\ w_2^1\ \dots\ w_{n_1}^1$
[SEP] $w_1^2\ w_2^2\ \dots\ w_{n_2}^2$
[SEP] $w_1^3\ w_2^3\ \dots\ w_{n_3}^3$
[SEP][PAD][PAD][PAD]
$z_{CLS}\ z_1^1\ z_2^1\ \dots\ z_{n_1}^1$
$z_{SEP}^1 z_1^2\ z_2^2\ \dots\ z_{n_2}^2$
$z_{SEP}^2 z_1^3\ z_2^3\ \dots\ z_{n_3}^3$
$z_{SEP}^3 z_{PAD} z_{PAD}\ z_{PAD}$
r_i
$z_{(e_1, r_i, e_2)}$
Tuple Classifier
p_i

Figure 1: Model architecture for *PARE*. Entity markers not shown for brevity.

lease our code for reproducibility.[1] We believe that our work represents a simple but strong baseline that can form the basis for further DS-RE research.

2 Related Work

Monolingual DS-RE: Early works in DS-RE build probabilistic graphical models for the task (e.g., (Hoffmann et al., 2011; Ritter et al., 2013). Most later works follow the multi-instance multi-label learning framework (Surdeanu et al., 2012) in which there are multiple labels associated with a bag, and the model is trained with at-least-one assumption. Most neural models for the task encode each sentence separately, e.g., using Piecewise CNN (Zeng et al., 2015), Graph Convolution Net (e.g., *RESIDE* (Vashishth et al., 2018)), GPT (*DISTRE* (Alt et al., 2019)) and BERT (*REDSandT* (Christou and Tsoumakas, 2021), *CIL* (Chen et al., 2021)). They all aggregate embeddings using intra-bag attention (Lin et al., 2016). Beyond Binary Cross Entropy, additional loss terms include masked language model pre-training (*DISTRE*, *CIL*), RL loss (Qin et al., 2018), and auxiliary contrastive learning (*CIL*). We show that *PARE* is competitive with *DISTRE*, *RESIDE*, *CIL*, and other natural baselines, without using additional pre-training, side information or auxiliary losses during training, unlike some comparison models.

To evaluate DS-RE, at test time, the model makes a prediction for an unseen bag. Unfortunately, most popular DS-RE dataset (NYT-10d) has a noisy test set, as it is automatically annotated (Riedel et al., 2010). Recently Gao et al. (2021) has released NYT-10m and Wiki-20m, which have manually annotated test sets. We use all three datasets in our work.

Multilingual DS-RE: A bilingual DS-RE model named MNRE (tested on English and Mandarin) introduced cross-lingual attention in language-specific CNN encoders (Lin et al., 2017). Recently, Bhartiya et al. (2022) has released a dataset, DiS-ReX, for four languages – English, Spanish, German and French. We compare *mPARE* against the state of the art on DiS-ReX, which combines MNRE architecture with mBERT encoder. See Appendix E for details on all DS-RE models.

Passage Construction from Bag of Sentences: At a high level, our proposed model builds a passage by combining the sentences in a bag that mentions a given entity pair. This idea of passage construction is related with the work of Yan et al. (2020), but with important differences, both in task definitions and neural models. First, they focus on predicting the tail entity of a given query $(e_1, r, ?)$, whereas our goal is relation prediction given an entity pair. There are several model differences such as in curating a passage, in use of trainable query vectors for relations, in passage construction strategy, etc. Importantly, their architecture expects a natural language question for each candidate relation – not only this requires an additional per-relation annotation (that might not be feasible for datasets having too many relations in the ontology), but also, it makes their method slower, since separate forward passes are needed per relation.

3 Passage Attended Relation Extraction

PARE explores the value of cross-sentence attention during encoding time. It uses a sequence of three key steps: passage construction, encoding and summarization, followed by prediction. Figure 1 illustrates these for a three-sentence bag.

Passage Construction constructs a *passage* $P(e_1, e_2)$ from sentences $t \in B(e_1, e_2)$. The construction process uses a sequential sampling of sentences in the bag without replacement. It terminates if (a) adding any new sentence would exceed the maximum number of tokens allowed by the encoder (512 tokens for BERT), or (b) all sentences from the bag have been sampled.

Passage Encoding takes the constructed passage and sends it to an encoder (BERT or mBERT) to generate contextualized embeddings $\mathbf{z}_j$ of every

[1] https://github.com/dair-iitd/DSRE

Model	AUC	P@M
PCNN-Att	34.1	69.4
RESIDE	41.5	77.2
DISTRE	42.2	66.8
REDSandT	42.4	75.3
CIL	50.8	86.0
PARE	53.4	84.8

Model	NYT-10m		Wiki-20m	
	AUC	M-F1	AUC	M-F1
B+Att	51.2	25.8	70.9	64.3
B+Avg	56.7	35.7	89.9	82.0
B+One	58.1	33.9	88.9	81.1
CIL	59.4	36.3	89.7	82.6
PARE	62.2	38.4	91.4	83.9

Model	AUC	μF1	M-F1
PCNN+Att	67.8	63.4	43.7
mB+Att	80.6	74.1	69.9
mB+One	80.9	74.0	68.9
mB+Avg	82.4	75.3	71.0
mB+MNRE	82.1	76.1	72.7
mPARE	87.0	79.3	76.0

Table 1: Results on (a) NYT-10d, (b) NYT-10m & Wiki-20m, and (c) DiS-ReX. B=BERT and mB=mBERT. *PARE* and *mPARE* outperforms all models by statistically significant margins (McNemar's test): all p values $< 10^{-5}$.

token w_j in the passage. For this, it first creates an encoder input. The input starts with the [CLS] token, followed by each passage sentence separated by [SEP], and pads all remaining tokens with [PAD]. Moreover, following best-practices in RE (Han et al., 2019), each mention of e_1 and e_2 in the passage are surrounded by special entity marker tokens <e1>,</e1>, and <e2>,</e2>, respectively.

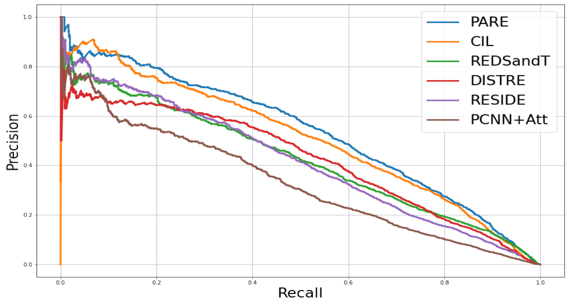
Passage Summarization maintains a (randomly-initialized) query vector $\mathbf{r}_i$ for every relation r_i. It then computes α^i_j, the normalized attention of r_i on each token w_j, using dot-product attention. Finally, it computes a relation-attended summary of the whole passage $\mathbf{z}_{(e_1,r_i,e_2)} = \sum_{j=1}^{j=L} \alpha^i_j \mathbf{z}_j$, where L is the input length. We note that this summation also aggregates embeddings of [CLS], [SEP], [PAD], as well as entity marker tokens.

Tuple Classifier passes $\mathbf{z}_{(e_1,r_i,e_2)}$ through an MLP followed by Sigmoid activation to return the probability p_i of the triple (e_1, r_i, e_2). This MLP is shared across all relation classes. At inference, a positive prediction is made if $p_i >$ threshold (0.5).

Loss Function is simply Binary Cross Entropy between gold and predicted label set for each bag. No additional loss terms are used.

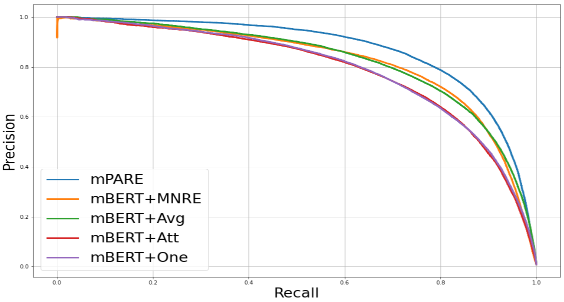
4 Experiments and Analysis

Figure 2: PR Curve for Models on NYT-10d



We compare *PARE* and *mPARE* against the state of the art models on the respective datasets. We also perform ablations and analyses to understand model behavior and reasons for its performance.

Figure 3: PR Curve for Models on DiS-ReX



Datasets and Evaluation Metrics: We evaluate *PARE* on three English datasets: NYT-10d, NYT-10m, Wiki-20m. *mPARE* is compared using the DiS-ReX benchmark. Data statistics are in Table 2, with more details in Appendix C. We use the evaluation metrics prevalent in literature for each dataset. These include AUC: area under the precision-recall curve, M-F1: macro-F1, μ-F1: micro-F1, and $P@M$: average of P@100, P@200 and P@300, where P@k denotes precision calculated over a model's k most confidently predicted triples.

Comparison Models and Hyperparameters: Since there is substantial body of work on NYT-10d, we compare against several recent models: *RESIDE*, *DISTRE*, *REDSandT* and the latest state of the art, *CIL*. For NYT-10m and Wiki-20m, we report comparisons against models in the original paper (Gao et al., 2021), and also additionally run CIL for a stronger comparison. For DiS-ReX, we compare against mBERT based models. See Appendix E for more details on the baseline models. For *PARE* and *mPARE*, we use base-uncased checkpoints for BERT and mBERT, respectively. Hyperparameters are set based on a simple grid search over devsets. (see Appendix A).

Dataset	#Rels	#Total	#Test	Test set
NYT-10d	58	694k	172k	Distant Sup.
NYT-10m	25	474k	9.74k	Manual
Wiki-20m	81	901k	140k	Manual
DiS-ReX	37	1.84M	334k	Distant Sup.

Table 2: Dataset statistics.

4.1 Comparisons against State of the Art

The results are presented in Table 1, in which, the best numbers are highlighted and second best numbers are underlined. On NYT-10d (Table 1(a)), *PARE* has 2.6 pt AUC improvement over *CIL*, the current state of the art, while achieving slightly lower P@M. This is also reflected in the P-R curve (Figure 2), where in the beginning our P-R curve is slightly on the lower side of CIL, but overtakes it for higher threshold values of recall. Our model beats *REDSandT* by 11 AUC pts, even though both use BERT, and latter uses extra side-information (e.g., entity-type, sub-tree parse).

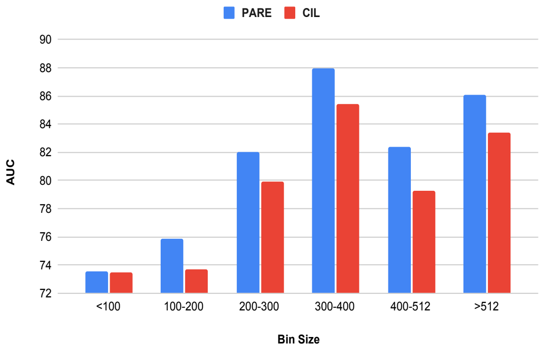
On manually annotated testsets (Table 1(b)), *PARE* achieves up to 2.8 pt AUC and 2.1 pt macro-F1 gains against *CIL*. We note that Gao et al. (2021) only published numbers on simpler baselines (BERT followed by attention, average and max aggregators, the details for which can be found in Appendix E), which are substantially outperformed by *PARE*. *CIL*'s better performance is likely attributed to its contrastive learning objective – it will be interesting to study this in the context of *PARE*.

For multilingual DS-RE (Table 1(c)), *mPARE* obtains a 4.9 pt AUC gain against mBERT+MNRE. P-R curve in Figure 3 shows that it convincingly outperforms others across the entire domain of recall values. We provide language-wise and relation-wise metrics in Appendix L – the gains are consistent on all languages and nearly all relations.

4.2 Analysis and Ablations

Generalizing to Unseen KB: Recently, Ribeiro et al. (2020) has proposed a robustness study in which entity names in a bag are replaced by other names (from the same type) to test whether the extractor is indeed reading the text, or is simply overfitting on the regularities of the given KB. We also implement a similar robustness study (details in Appendix K), where entity replacement results in an entity-pair bag that does not exist in the original KB. We find that on this modified NYT-10m, all models suffer a drop in performance, suggesting that models are not as robust as we intend them to be. We, however, note that *CIL* suffers a 28.1% drop in AUC performance, but *PARE* remains more robust with only a 16.8% drop. We hypothesize that this may be because of *PARE*'s design choice of attending on all words for a given relation, which could reduce its focus on entity names themselves.

Figure 4: AUC on different bins of the NYT-10m test set. The x-axis denotes the range of lengths of untruncated passages in each bin



Scaling with Size of Entity-Pair Bags: Due to truncation when the number of tokens in a bag exceed 512 (limit for BERT), one would assume that *PARE* may not be suited for cases where the number of tokens in a bag is large. To study this, we divide the test set of NYT-10m into 6 different bins based on the number of tokens present in the untruncated passage (details on the experiment in Appendix J). We present results in Figure 4. We find that *PARE* shows consistent gains of around 2 to 3 pt in AUC against *CIL* for all groups except the smallest group. This is not surprising, since for smallest group, there is likely only one sentence in a bag, and *PARE* would not gain from inter-sentence attention. For large bags, relevant information is likely already present in truncated passage, due to redundancy.

Attention Patterns: In *PARE*, each relation class has a trainable query vector, which attends on every token. The attention scores could give us some insight about the words the model is focusing on. We observe that for a candidate relation that is not a gold label for a particular bag, surprisingly, the highest attention scores are obtained by [PAD] tokens. In fact, for such bags, on an average, roughly 90% of the attention weight goes to [PAD] tokens, whereas this number is only 0.1% when the relation is in the gold set (see Appendices H and I). We find this to be an example of model ingenu-

Modification	Change in AUC
w/o passage summarization	-4.9
w/o [PAD] attention	-3.1
w/o entity markers	-36.9

Table 3: Change in AUC on NYT-10d by removing various architectural components from *PARE*

ity – *PARE* seems to have creatively learned that whenever the most appropriate words for a relation are not present, it could simply attend on [PAD] embeddings, which may lead to similar attended summaries, which may be easily decoded to a low probability of tuple validity. In fact, as a further test, we perform an ablation where we disallow relation query vectors to attend on [PAD] tokens – this results in an over 3 pt drop in AUC on NYT-10d, indicating the importance of padding for prediction (see Table 3).

Ablations: We perform further ablations of the model by removing entity markers and removing the relation-attention step that computes a summary (instead using [CLS] token for predicting each relation). *PARE* loses significantly in performance in each ablation obtaining 16.5 and 48.5 AUC, respectively (as against 53.4 for full model) on NYT-10d (table 3). The critical importance of entity markers is not surprising, since without them the model does not know what is the entity-pair it is predicting for. We also notice a very significant gain due to relation attention and passage summarization, suggesting that this is an important step for the model – it allows focus on specific words relevant for predicting a relation. We perform the same experiments on the remaining datasets and observe similar results (Appendix G).

Effect of Sentence Order: We build 20 random passages per bag (by varying sentence order and also which sentences get selected if passage needs truncation). On all four datasets (Appendix M), we find that the standard deviation to be negligible. This analysis highlights 1) the sentence-order invariance of *PARE*'s performance and 2) In practical settings, the randomly sampled sentences with token limit of 512 in the passage is good enough to make accurate bag-level predictions.

5 Conclusion and Future Work

We introduce *PARE*, a simple baseline for the task of distantly supervised relation extraction. Our experiments demonstrate that this simple baseline produces very strong results for the task, and outperforms existing top models by varying margins across four datasets in monolingual and multilingual settings. Several experiments for studying model behavior show its consistent performance that generalizes across settings. We posit that our framework would serve as a strong backbone for further research in the field of DS-RE.

There are several directions to develop the *PARE* architecture further. E.g., *PARE* initializes relation embeddings randomly and also constructs passage via random sampling. Alternatively, one could make use of label descriptions and aliases from Wikidata to initialize label query vectors; one could also use a sampling strategy to filter away noisy sentences (e.g. a sentence selector (Qin et al., 2018) module integrated with *PARE*). In the multilingual setting, contextualized embeddings of entity mentions in a passage may be aligned using constrained learning techniques (Mehta et al., 2018; Nandwani et al., 2019) to learn potentially better token embeddings. Constraints can be imposed on the label hierarchy as well (E.g. *PresidentOf* ⇒ *CitizenOf*, etc.) since label query vectors operate independently of each other on the passage in *PARE*. Additionally, translation-based approaches at training or inference (Nag et al., 2021; Kolluru et al., 2022) could improve *mPARE* performance. Recent ideas of joint entity and relation alignment in multilingual KBs (Singh et al., 2021) may be combined along with *mPARE*'s relation extraction capabilities.

Acknowledgements

This work is primarily supported by a grant from Huwaei. Mausam is supported by grants from Google and Jai Gupta Chair Professorship. Parag was supported by the DARPA Explainable Artificial Intelligence (XAI) Program (#N66001-17-2-4032). Mausam and Parag are supported by IBM SUR awards. Vipul is supported by Prime Minister's Research Fellowship (PMRF). We thank IIT Delhi HPC facility for compute resources. We thank Abhyuday Bhartiya for helping in reproducing results from the DiS-ReX paper, and Keshav Kolluru for helpful comments on an earlier draft of the paper. Any opinions, findings, conclusions or recommendations expressed here are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of the funding agencies.

References

Christoph Alt, Marc Hübner, and Leonhard Hennig. 2019. Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1388–1398, Florence, Italy. Association for Computational Linguistics.

Abhyuday Bhartiya, Kartikeya Badola, and Mausam. 2022. DiS-ReX: A multilingual dataset for distantly supervised relation extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, Dublin, Ireland. Association for Computational Linguistics.

Tao Chen, Haizhou Shi, Siliang Tang, Zhigang Chen, Fei Wu, and Yueting Zhuang. 2021. CIL: Contrastive instance learning framework for distantly supervised relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6191–6200, Online. Association for Computational Linguistics.

Despina Christou and Grigorios Tsoumakas. 2021. Improving distantly-supervised relation extraction through bert-based label and instance embeddings. *IEEE Access*, 9:62574–62582.

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

Tianyu Gao, Xu Han, Yuzhuo Bai, Keyue Qiu, Zhiyu Xie, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Manual evaluation matters: Reviewing test protocols of distantly supervised relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1306–1318, Online. Association for Computational Linguistics.

Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Jie Zhou, and Maosong Sun. 2020. More data, more relations, more context and more openness: A review and outlook for relation extraction. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 745–758, Suzhou, China. Association for Computational Linguistics.

Xu Han, Tianyu Gao, Yuan Yao, Deming Ye, Zhiyuan Liu, and Maosong Sun. 2019. OpenNRE: An open and extensible toolkit for neural relation extraction. In *Proceedings of EMNLP-IJCNLP: System Demonstrations*, pages 169–174.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 541–550, Portland, Oregon, USA. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Keshav Kolluru, Mohammed Muqeeth, Shubham Mittal, Soumen Chakrabarti, and Mausam. 2022. Alignment-Augmented Consistent Translation for Multilingual Open Information Extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, Dublin, Ireland. Association for Computational Linguistics.

Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2017. Neural relation extraction with multi-lingual attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 34–43, Vancouver, Canada. Association for Computational Linguistics.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133, Berlin, Germany. Association for Computational Linguistics.

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

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Sanket Vaibhav Mehta, Jay Yoon Lee, and Jaime G Carbonell. 2018. Towards semi-supervised learning for deep semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4958–4963.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Arijit Nag, Bidisha Samanta, Animesh Mukherjee, Niloy Ganguly, and Soumen Chakrabarti. 2021. A data bootstrapping recipe for low-resource multilingual relation classification. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 575–587.

Yatin Nandwani, Abhishek Pathak, Mausam, and Parag Singla. 2019. A primal dual formulation for deep learning with constraints. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12157–12168.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018. Robust distant supervision relation extraction via deep reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2137–2147. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of nlp models with checklist. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Alan Ritter, Luke Zettlemoyer, Mausam, and Oren Etzioni. 2013. Modeling missing data in distant supervision for information extraction. *Trans. Assoc. Comput. Linguistics*, 1:367–378.

Harkanwar Singh, Soumen Chakrabarti, Prachi Jain, Sharod Roy Choudhury, and Mausam. 2021. Multilingual knowledge graph completion with joint relation and entity alignment. In *3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, October 4-8, 2021*.

Livio Baldini Soares, Nicholas Fitzgerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465, Jeju Island, Korea. Association for Computational Linguistics.

Shikhar Vashishth, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. 2018. RESIDE: Improving distantly-supervised neural relation extraction using side information. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1266, Brussels, Belgium. Association for Computational Linguistics.

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

Lingyong Yan, Xianpei Han, Le Sun, Fangchao Liu, and Ning Bian. 2020. From bag of sentences to document: Distantly supervised relation extraction via machine reading comprehension. *arXiv preprint arXiv:2012.04334*.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1753–1762.

A Experimental Settings

We train and test our model on two NVIDIA GeForce GTX 1080 Ti cards. We use a linear LR scheduler having weight decay of 1e-5 with AdamW (Loshchilov and Hutter, 2019; Kingma and Ba, 2015) as the optimizer. Our implementation uses PyTorch (Paszke et al., 2019), the Transformers library (Wolf et al., 2020) and OpenNRE [2] (Han et al., 2019). We use bert-base-uncased checkpoint for BERT initialization in the mono-lingual setting. For multi-lingual setting, we use bert-base-multilingual-uncased.

For hyperparameter tuning, we perform grid search over {1e-5, 2e-5} for learning rate and {16, 32, 64} for batch size and select the best performing configuration for each dataset.

PARE takes 2 epochs to converge on NYT-10d (152 mins/epoch), 3 epochs for NYT-10m (138 mins/epoch), 2 epochs for Wiki-20m (166 mins/epoch) and 4 epochs for DiS-ReX (220 mins/epoch).

The numbers we report for the baselines come from their respective papers. We obtained the code base of CIL, BERT+Att, BERT+Avg, BERT+One from their respective authors, so that we could run them on additional datasets. We were able to replicate same numbers as reported in their papers. We trained those models on other datasets as well by carefully tuning the bag size hyperparameter.

B Sizes of different models

We report the number of additional trainable parameters, in each model, on top of the underlying BERT/mBERT encoder (all models except MNRE use the bert-base-uncased checkpoint, whereas MNRE uses the bert-base-multilingual-uncased checkpoint) in table 4. We note that the key reason why *PARE* has significantly lower number of additional parameters (on top of the BERT/mBERT encoder) is because all the other models use *entity pooling* (Soares et al., 2019) for constructing instance representations. The *entity pooling* operation requires an additional fully-connected layer which projects the concatenated encoded representations of head and tail entity in an input instance to a vector of the same size (for BERT/mBERT, this results in additional $(2 \times 768)^2$ weight and 2×768 bias parameters).

Model	**#Parameters (excluding BERT)**
Att	2400793
One	2399257
Avg	2399257
CIL	2453052
MNRE	2645029
PARE	46082

Table 4: Comparison of trainable parameters between our model and other state-of-the-art models

C Dataset Details

We evaluate our proposed model on four different datasets: NYT-10d (Riedel et al., 2010), NYT-10m (Gao et al., 2021), Wiki-20m (Gao et al., 2021) and DiS-ReX (Bhartiya et al., 2022). The statistics for each of the datasets is present in table 2.

NYT-10d

NYT-10d is the most popular dataset for monolingual DS-RE, constructed by aligning Freebase entities to the New York Times Corpus. The train and test splits are both distantly supervised.

NYT-10m

NYT-10m is a recently released dataset to train and evaluate models for monolingual DS-RE. The dataset is built from the same New York Times Corpus and the Freebase KB but with a new relation ontology and a manually annotated test set. It aims to tackle the existing problems with the NYT-10d dataset by 1)

[2] https://github.com/thunlp/OpenNRE

establishing a public validation set 2) establishing consistency among the relation classes present in the train and test set 3) providing a high quality, manually labeled test set.

Wiki-20m

Wiki-20m is also a recently released dataset for training DS-RE models and evaluating them on manually annotated a test set. The test set in this case corresponds to the Wiki80 dataset (Han et al., 2019). The relation ontology of Wiki80 is used to re-structure the Wiki20 DS-RE dataset (Han et al., 2020), from which the training and validation splits are created. It is made sure that their is no overlap between the instances present in the testing and the training and validation sets.

DiS-ReX

DiS-ReX is a recently released benchmarking dataset for training and evaluating DS-RE models on instances spanning multiple languages. The entities present in this dataset are linked across the different languages which means that a bag can contain sentences from more than one languages. We use the publicly available train, validation and test splits and there is no overlap between the bags present in any two different dataset splits.
We obtain the first three datasets from OpenNRE and DiS-ReX from their official repository.

D Description of Intra-Bag attention

Let $t_1, t_2, ..., t_n$ denote n instances sampled from $B(e_1, e_2)$. In all models using intra-bag attention for instance-aggregation, each t_i is independently encoded to form the instance representation, $E(t_i)$, following which the relation triple representation B_r for the triple (e_1, e_2, r) is given by $B_r = \sum_{i=0}^{i=n} \alpha_i^r E(t_i)$. Here r is any one of the relation classes present in the dataset and α_i^r is the normalized attention score allotted to instance representation $E(t_i)$ by relation query vector $\overrightarrow{r}$ for relation r. The model then predicts whether the relation triple is a valid one by sending each B_r through a feed-forward neural network. In some variants, $\overrightarrow{r}$ is replaced with a shared query vector for all relation-classes, $\overrightarrow{q}$, resulting in a bag-representation B corresponding to (e_1, e_2) as opposed to triple-representation.

E Baselines

The details for each baseline is provided below:

PCNN-Att

Lin et al. (2016) proposed the intra-bag attention aggregation scheme in 2016, obtaining the then state-of-the-art performance on NYT-10d using a piecewise convolutional neural network (PCNN (Zeng et al., 2015)).

RESIDE

Vashishth et al. (2018) proposed RESIDE which uses side-information (in the form of entity types and relational aliases) in addition to sentences present in the dataset. The model uses intra-bag attention with a shared query vector to combine the representations of each instance in the bag. The sentence representations are obtained using a Graph Convolutional Network (GCN) encoder.

DISTRE

Alt et al. (2019) propose the use of a pre-trained transformer based language model (OpenAI GPT Radford et al. (2018)) for the task of DS-RE. The model uses intra-bag attention for the instance aggregation step.

REDSandT

Christou and Tsoumakas (2021) propose the use of a BERT encoder for DS-RE by using sub-tree parse of the input sentence along with special entity type markers for the entity mentions in the text. The model uses intra-bag attention for the instance aggregation step.

CIL
Chen et al. (2021) propose the use of Masked Language Modeling (MLM) and Contrastive Learning (CL) losses as auxilliary losses to train a BERT encoder + Intra-bag attention aggregator for the task.

BERT+Att/mBERT+Att
The model uses intra-bag attention aggregator on top of a BERT/mBERT encoder.

BERT+Avg/mBERT+Avg
The model uses "Average" aggregator which weighs each instance representation uniformly, hence denoting bag-representation as the average of instance-representations.

BERT+One/mBERT+One
The model independently performs multi-label classification on each instance present in the bag and then aggregates the classification results by performing class-wise max-pooling (over sentence scores). In essence, the "One" aggregator ends up picking one instance for each class (the one which denotes the highest confidence for that particular class), hence the name.

mBERT+MNRE
The MNRE aggregator was originally introduced by Lin et al. (2017) and used with a shared mBERT encoder by Bhartiya et al. (2022) [3]. The model assigns a query vector for each $(relation, language)$ tuple. A bag is divided into sub-bags where each sub-bag contains the instances of the same language. In essence, a bag has L sub-bags and each relation class corresponds to L query vectors, where L denotes the number of languages present in the dataset. These are then used to construct L^2 triple representations (using intra-bag attention aggregation on each *(sub-bag,query vector)* pair for a candidate relation) which are then scored independently. The final confidence score for a triple is the average of L^2 triple scores.

F Statistical Significance

We compare the predictions of our model on the non-NA triples present in the test set with the predictions of the second-best model using the McNemar's test of statistical significance (McNemar, 1947). In all cases, we obtained the *p-value* to be many orders of magnitude smaller than 0.05, suggesting that the improvement in results is statistically significant in all cases.

G Ablation Study

Modification	NYT-10d	NYT-10m	Wiki-20m	DiS-ReX
w/o passage summarization	-4.9	-2.9	-4.2	-0.8
w/o [PAD] attention	-3.1	-2.3	-1.9	-0.1
w/o entity markers	-36.9	-16.5	-29.9	-20.5

Table 5: Model ablation i.e. change in AUC performance with different components of *PARE*

We perform ablation studies on various datasets to understand which components are most beneficial for our proposed model. We provide the results in table 5.

We observe that upon replacing our passage summarization step with multi-label classification using [CLS] token (present at the start of the passage), we observe a significant decrease in AUC, indicating that contextual embedding of [CLS] token might not contain enough information for multi-label prediction of bag.

[3] Obtained from the original repository for DiS-ReX

For NYT-10, it is interesting to note here that the AUC is still higher than that of REDSandT, a model which uses BERT+Att as the backbone (along with other complicated machinery). This means that one can simply obtain an improvement in performance by creating a passage from multiple instances in a bag.

Removing entity markers resulted in the most significant drop in performance. However, this is also expected since without them, our model would have no way to understand which entities to consider while performing relation extraction.

H Attention on [PAD] tokens

In the passage summarization step (described in section 3), we allow the relation query vector $\overrightarrow{r}$ to also attend over the encodings of the [PAD] tokens present in the passage. We make this architectural choice in-order to provide some structure to the relation-specific summaries created by our model. If a particular relation class r is not a valid relation for entity pair (e_1, e_2), then ideally, we would want the attended-summary of the passage $P(e_1, e_2)$ created by the relation vector $\overrightarrow{r}$ to represent some sort of a null state (since information specific to that relation class is not present in the passage). Allowing [PAD] tokens to be a part of the attention would provide enough flexibility to the model to represent such a state. We test our hypothesis by considering 1000 non-NA bags correctly labelled by our trained model in the test set of NYT-10d. Let $R(e_1, e_2)$ denote the set of valid relation-classes for entity pair (e_1, e_2) and let R denote all of the relation-classes present in the dataset. We first calculate the percentage of attention given to [PAD] tokens for a given passage $P(e_1, e_2)$ for all relation-classes in R. The results are condensed into two scores, sum of scores for $R(e_1, e_2)$ and sum of scores for $R \setminus R(e_1, e_2)$. The results are aggregated for all 1000 bags, and then averaged out by dividing with the total number of positive triples and negative triples respectively. We obtain that on an average, only 0.07% of attention weight is given to [PAD] tokens by relation vectors corresponding to $R(e_1, e_2)$, compared to 88.35% attention weight given by relation vectors corresponding to $R \setminus R(e_1, e_2)$. We obtain similar statistics on other datasets as well. This suggests that for invalid triples, passage summaries generated by the model resemble the embeddings of the [PAD] token. Furthermore, since we don't allow [PAD] tokens to be a part of self-attention update inside BERT, the [PAD] embeddings at the output of the BERT encoder are not dependent on the passage, allowing for uniformity across all bags.

Finally, we train a model where we don't allow the relation query vectors to attend on the [PAD] token embeddings and notice a 3.1pt drop in AUC on NYT-10d (table 5). We also note that the performance is still significantly higher than models such as REDSandT and DISTRE, suggesting that our instance aggregation scheme still performs better than the baselines, even when not optimized fully.

I Examples of Attention Weighting during Passage Summarization

To understand how the query vector of a relation attends over passage tokens to correctly predict that relation, we randomly selected from correctly predicted non-NA triples and selected the token obtaining the highest attention score (by the query vector for the correct relation). For the selection, we ignore the stop words, special tokens and the entity mentions. The results are presented in table 6.

J Performance vs Length of test passages

Our instance aggregation scheme truncates the passage if the number of tokens exceed the maximum number of tokens allowed by the encoder. In such cases, one would assume that the our model is not suited for cases where the number of instances present in a bag is very large. To test this hypothesis, we divide the non-NA bags, (e_1, e_2), present in the NYT-10m data into 6 bins based on the number of tokens present in $P(e_1, e_2)$ (after tokenized using BERT). We then compare the performance with CIL on examples present in each bin. The results in figure 4 indicate that a) our model beats CIL in each bin-size b) the performance trend across different bins is the same for both models. This trend is continued even for passages where the number of tokens present exceed the maximum number of tokens allowed for BERT (i.e. 512). This results indicate that 512 tokens provide sufficient information for correct classification of a triple. Moreover, models using intra-bag attention aggregation scheme fix the number of instances sampled from the bag in practice. For CIL, the best performing configuration uses a bag-size of 3. This

Input Passage (tokenized by BERT)	**correctly predicted label**
[CLS] six months later , his widow met the multi ##mill ##ion ##aire [unused2] vincent astor [unused3] , a **descendant** of the fur trader turned manhattan real - estate magnate [unused0] john jacob astor [unused1] , and a man considered so unpleasant by his peers 1 ##rb and even by his own mother rr ##b - that he reportedly required a solitary seating for lunch at his club because nobody would share a meal with him . [SEP]	/people/person/children
[CLS] the [unused2] robin hood foundation [unused3] , **founded** by [unused0] paul tudor jones [unused1] ii and perhaps the best - known hedge fund charity , raised $ 48 million at its annual benefit dinner last year . [SEP]	/business/person/company
[CLS] she is now back in the fourth round , where she will face 11th - seeded je ##lena jan ##kovic of serbia , a 6 - 3 , 6 - 4 winner over [unused0] victoria az ##are ##nka [unused1] **of** [unused2] belarus [unused3] . [SEP]	/people/person/nationality
[CLS] [unused2] boston [unused3] what : a two - bedroom condo how much : $ 59 ##9 , 000 per square foot : $ 83 ##6 located in the [unused0] back bay [unused1] area of the city , this 71 ##6 - square - foot condo has views from the apartment and its private roof deck of the charles river , one block away . [SEP] seven years ago , when nad ##er tehran ##i and monica ponce de leon , partners at office da , an architecture firm in [unused2] boston [unused3] , were asked to reno ##vate a five - story town house in the [unused0] back bay [unused1] **neighborhood** , they faced a singular design challenge . [SEP] far more inviting is first church in [unused2] boston [unused3] , in [unused0] back bay [unused1] , which replaced a gothic building that burned in 1968 . [SEP]	/location/neighborhood/neighborhood_of
[CLS] [unused0] michael sm ##uin [unused1] , a choreographer who worked for major ballet companies and led his own , marshal ##ing eclectic dance forms , robust athletic ##ism and striking theatrical ##ity to create works that appealed to broad audiences , **died** yesterday in [unused2] san francisco [unused3] . [SEP]	/people/deceasedperson/place_of_death
[CLS] [unused2] steve new ##comb [unused3] , a [unused0] powers ##et [unused1] **founder** and veteran of several successful start - ups , said his company could become the next google . [SEP]	/business/company/founders

Table 6: Attention analysis on a few random correctly predicted non-NA triples on NYT-10m test set. The highest attention-scored token (excluding entity mentions and special markers and stop words) are present in bold. [unused0], [unused1] denote the start and end head entity markers. [unused2], [unused3] denote the start and end tail entity markers.

analysis therefore indicates that our model doesn't particularly suffer a drop in performance on large bags when compared with other state-of-the-art models.

K Entity Permutation Test

To understand how robust our trained model would be to changes in the KB, we design the entity permutation test (inspired by Ribeiro et al. (2020)). An ideal DS-RE model should be able to correctly predict the relationship between an entity pair by understanding the semantics of the text mentioning them. Since DS-RE models under the multi-instance multi-label (Surdeanu et al., 2012) (MI-ML) setting are evaluated on bag-level, it might be the case that such models are simply memorizing the KB on which they are being trained on.

To test this hypothesis, we construct a new test set (in fact, 5 such sets and report average over those 5) using NYT-10m by augmenting its KB. Let $B(e_1, e_2)$ denote a non-NA bag already existing in the test set of the dataset. We augment this bag to correspond to a new entity-pair (which is not present in the combined KB of all three splits of this dataset). The augmentation can be of two different types: replacing e_1 with e'_1 or replacing e_2 with e'_2. We restrict such augmentations to the same type (i.e the type of e_i and e'_i is same for $i = 1, 2$). For each non-NA entity pair in the test set of the dataset, we select one such augmentation and appropriately modify each instance in $B(e_1, e_2)$ to have the new entity mentions. We note that since each instance in NYT-10m is manually annotated and since our augmentation ensures that the type signature is preserved, the transformation is label preserving. For the NA bags, we use the ones already present in the original split. This entire transformation leaves us with an augmented test set, having same number of NA and non-NA bags as the original split. The non-NA entity pairs are not present in the KB on which the model is trained on.

L More Analysis on DiS-ReX

L.1 Relation-wise F1 scores

To show how our model performs on each relation label compared to other competitive baselines, we present relation-wise F1 scores on DiS-ReX in table 7.

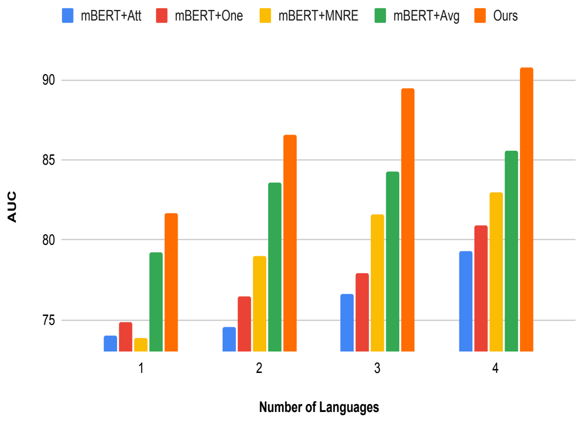
L.2 Language-wise AUC scores

We compare the performance of our model compared to other baselines on every language in DiS-ReX. For this, we partition the test data into language-wise test sets i.e. containing instances of only a particular language. The results are presented in table 8. We observe that the order of performance across languages is consistent for all models including ours i.e. German < English < Spanish < French. Further we observe that our model beats the second best model by an AUC ranging from 3 upto 4 points on all languages.

L.3 Do multilingual bags improve performance?

To understand whether the currently available aggregation schemes (including ours) are able to benefit from multilingual bags or not, we conduct an experiment where we only perform inference on test-set bags that contain instances from all four languages. In the multilingual case, the *passage* constructed during the *Passage Summarization* step will contain multiple sentences of different languages. To understand whether such an input allows improves (or hampers) the performance, we devise an experiment where we perform inference by removing sentences from any one, two or three languages from the set of bags containing instances of all four languages. There are roughly 1500 bags of such kind. Note that removing any k languages ($k <= 3$) would result in $\binom{4}{k}$ different sets and we take average of AUC while reporting the numbers. The results are presented in figure 5.

Figure 5: AUC vs number of languages in a bag in DiS-ReX test set



We observe that in all aggregation schemes, AUC increases with increase in number of languages of a multilingual bag. *mPARE* consistently beats the other models in each scenario, indicating that the encoding of a multilingual passage and attention-based summarization over multilingual tokens doesn't hamper the performance of a DS-RE model with increasing no. of languages.

Relation	***mPARE***	**mBERT-MNRE**	**mBERT-Avg**
http://dbpedia.org/ontology/birthPlace	**77.5**	75.3	74.9
http://dbpedia.org/ontology/associatedBand	**77.9**	70.9	74.7
http://dbpedia.org/ontology/director	**88.4**	83.2	85.5
http://dbpedia.org/ontology/country	**88.4**	86	85.2
http://dbpedia.org/ontology/deathPlace	**71.0**	67.3	65.5
http://dbpedia.org/ontology/nationality	**70.4**	67.7	68.7
http://dbpedia.org/ontology/location	**74.2**	70.5	67.5
http://dbpedia.org/ontology/related	**78.9**	75.5	73.2
http://dbpedia.org/ontology/isPartOf	**74.8**	68.6	64.7
http://dbpedia.org/ontology/influencedBy	57.7	**58.4**	57.4
http://dbpedia.org/ontology/starring	**87.5**	86.1	83.9
http://dbpedia.org/ontology/headquarter	**74.0**	70.7	66.7
http://dbpedia.org/ontology/successor	**74.2**	71.8	71.3
http://dbpedia.org/ontology/bandMember	**76.2**	74.6	74.3
http://dbpedia.org/ontology/producer	**56.7**	53.6	48.5
http://dbpedia.org/ontology/recordLabel	**90.5**	86.9	86.1
http://dbpedia.org/ontology/city	**83.2**	78.8	77.6
http://dbpedia.org/ontology/influenced	56.3	**61.9**	51.5
http://dbpedia.org/ontology/author	**81.6**	78.2	80.5
http://dbpedia.org/ontology/team	**84.8**	82.5	78.6
http://dbpedia.org/ontology/formerBandMember	56.4	**57.4**	56.5
http://dbpedia.org/ontology/state	**86.9**	83.9	82.4
http://dbpedia.org/ontology/region	**84.8**	80.4	78.8
http://dbpedia.org/ontology/subsequentWork	**74.1**	72.4	69.6
http://dbpedia.org/ontology/department	**96.4**	95.4	95.5
http://dbpedia.org/ontology/locatedInArea	**76.4**	72.5	72.3
http://dbpedia.org/ontology/artist	**80.8**	77.2	78.6
http://dbpedia.org/ontology/hometown	**78.8**	73.6	73.7
http://dbpedia.org/ontology/province	**82.1**	79.2	78.2
http://dbpedia.org/ontology/riverMouth	**77.2**	72.4	71.9
http://dbpedia.org/ontology/locationCountry	**66.9**	62.5	64.2
http://dbpedia.org/ontology/predecessor	67.3	**68.1**	62
http://dbpedia.org/ontology/previousWork	68.6	**69.6**	65.5
http://dbpedia.org/ontology/capital	**68.6**	55.1	58
http://dbpedia.org/ontology/leaderName	**78.4**	70.4	63.3
http://dbpedia.org/ontology/largestCity	**65.7**	59.1	48.6

Table 7: Relation-wise F1 scores on DiS-Rex. Bold and underline represent best and second best models respectively on a class. Our model consistently beats the other 2 models in 31 out of 36 relation classes, thus showing how strong our approach is for the multilingual setting.

Model	**English**	**French**	**German**	**Spanish**
mPARE	**83.2**	**86.8**	**81.7**	**85.3**
mBERT-Avg	79.9	83.1	77.7	82.1
mBERT-MNRE	79.6	82.2	75.5	81.6

Table 8: Language-wise AUC comparison of our model v/s baseline models.

M Negligible effect of random ordering

Since we order the sentences randomly into a passage to be encoded by BERT, this may potentially cause some randomness in the results. However, we hypothesize that the BERT encoder must also be getting fine-tuned to treat the bag as a set (and not a sequence) of sentences when being trained with random ordering technique. And as a result, it's performance must be agnostic to the order of sentences it sees in a passage during inference. To validate this, we perform 20 inference runs of our trained model with different seeds i.e. the ordering of sentences is entirely random in each run. We measure mean and standard deviation for each dataset as listed in table 9. We observe negligible standard deviation in all metrics. A minute variation in Macro-F1 or P@M metrics may be attributed to the fact that these are macro-aggregated metrics and a variation in performance over some data points may also affect these to some extent.

	NYT-10m		**NYT-10d**		**Wiki-20m**		**DiS-ReX**	
	AUC	**M-F1**	**AUC**	**P@M**	**AUC**	**M-F1**	**AUC**	**M-F1**
	62.11	38.35	53.49	84.82	91.41	83.87	87.03	76.01
	62.11	38.44	53.43	84.72	91.41	83.88	87.06	76.18
	62.18	38.27	53.49	84.69	91.41	83.85	87.0	76.04
	62.11	38.32	53.45	84.56	91.42	83.88	86.98	75.93
	62.12	38.34	53.64	84.62	91.43	84.04	87.03	76.03
	62.25	38.46	53.6	84.73	91.42	83.82	87.04	76.07
	62.16	38.54	53.54	85.18	91.42	83.81	87.01	76.0
	62.2	38.68	53.45	84.57	91.41	83.91	86.99	75.98
	62.22	38.27	53.43	84.4	91.42	83.83	87.06	76.2
	62.19	38.47	53.47	84.68	91.41	83.81	87.02	76.06
	62.22	38.43	53.45	84.51	91.41	83.85	87.03	75.99
	62.13	38.4	53.5	85.18	91.41	83.85	87.06	76.14
	62.21	38.3	53.58	85.23	91.42	83.87	87.02	75.96
	62.18	38.15	53.4	84.51	91.43	83.91	87.01	75.97
	62.21	38.51	53.44	84.54	91.41	83.88	87.04	76.1
	62.2	38.34	53.53	84.51	91.41	83.91	87.03	76.04
	62.13	38.29	53.61	84.56	91.43	83.96	87.02	76.05
	62.23	38.63	53.46	84.79	91.41	83.81	87.04	76.13
	62.19	38.3	53.42	84.46	91.41	83.85	87.03	75.96
	62.29	38.36	53.47	85.07	91.42	83.87	87.01	76.01
Average	62.18	38.39	53.49	84.71	91.42	83.87	87.03	76.01
Std-Dev	0.05	0.13	0.07	0.25	0.01	0.06	0.01	0.07
Std-Dev(%)	0.08	0.34	0.13	0.3	0.01	0.07	0.01	0.1

Table 9: We perform 20 inference runs with random seeds of our trained model on each dataset and report the mean and standard deviation. All numbers have been rounded upto second decimal place. We observe negligible stdandard deviation in all metrics on all datasets thus validating our hypothesis that the model learns to treat a bag of sentences as a set (and not a sequence) of sentences treating any random order almost alike. Note that the results presented in main paper are for inference done with same seed value with which the model has been trained. However, in current analysis we select random seed values at inference (irrespective of the one with which it was trained).

To Find Waldo You Need Contextual Cues: Debiasing *Who's Waldo*

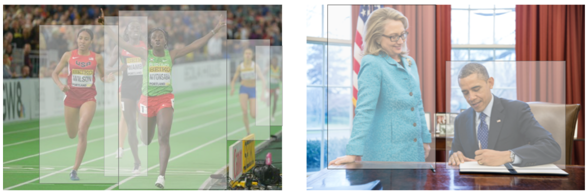
Yiran Luo Pratyay Banerjee Tejas Gokhale Yezhou Yang Chitta Baral
Arizona State University, Tempe, AZ, USA
{yluo97, pbanerj6, tgokhale, yz.yang, chitta}@asu.edu

Abstract

We present a debiased dataset for the Person-centric Visual Grounding (PCVG) task first proposed by Cui et al. (2021) in the *Who's Waldo* dataset. Given an image and a caption, PCVG requires pairing up a person's name mentioned in a caption with a bounding box that points to the person in the image. We find that the original *Who's Waldo* dataset compiled for this task contains a large number of biased samples that are solvable simply by heuristic methods; for instance, in many cases the first name in the sentence corresponds to the largest bounding box, or the sequence of names in the sentence corresponds to an exact left-to-right order in the image. Naturally, models trained on these biased data lead to over-estimation of performance on the benchmark. To enforce models being correct for the correct reasons, we design automated tools to filter and debias the original dataset by ruling out all examples of insufficient context, such as those with no verb or with a long chain of conjunct names in their captions. Our experiments show that our new sub-sampled dataset[1] contains less bias with much lowered heuristic performances and widened gaps between heuristic and supervised methods. We also demonstrate the same benchmark model trained on our debiased training set outperforms that trained on the original biased (and larger) training set on our debiased test set. We argue our debiased dataset offers the PCVG task a more practical baseline for reliable benchmarking and future improvements.

1 Introduction

A newly released task called Person-centric Visual Grounding (Cui et al., 2021) poses an interesting angle into contextual reasoning in vision-language. The task is motivated by humans' reasoning ability.



Women's 800 metres final at 2016 IAAF World Indoor Championships in Portland : [NAME], [NAME] and [NAME]

Secretary of State [NAME] watches as President [NAME] signs a Presidential memorandum

Figure 1: We find many biased data from the original Who's Waldo dataset contain insufficient contextual cues and cannot be used to map names to persons in an image. **Left:** An unsolvable example with no actions nor descriptions w.r.t the detected persons. Given no background knowledge about the individuals, one can only guess the masked [NAME]'s based on heuristic biases such as the locations of the bounding boxes. **Right:** A qualifying example with clearly worded interactions (e.g. detectable verbs such as 'watches' & 'signs') about each masked name - the very type of data we incorporate into our debiased dataset.

Humans viewing an image with a caption as shown in Figure 1 can reason (and if needed, speculate) which name refers to which person in the image. This reasoning task involves multiple abilities, such as perceiving characteristics and behaviors of people, understanding their actions in context, speculating about their intentions and effects human of actions (Fang et al., 2020), and connecting visually perceived characteristics with grounded descriptions in natural language (Kazemzadeh et al., 2014; Yu et al., 2016; Zellers et al., 2019). In many cases, this task can be performed without knowing the names of the people; for instance in the example on the right, one person is signing and the other is not, as such it is possible to predict which person refers to President and Secretary of State respectively. However, in cases such as the example on the left, if all persons are performing the same action (run-

[1] Available at: https://github.com/fpsluozi/tofindwaldo

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 355 - 361
May 22-27, 2022 ©2022 Association for Computational Linguistics

ning on a track), then it is hard to match names with these runners without any additional information. Progress in the PCVG task can thus help better capture what exact contextual cues are needed to learn about a person's characteristics in a scenario, and can aid improvements in visual understanding about human interactions and behaviors.

To support this task, Cui et al. (2021) offer a large-scale dataset called *Who's Waldo* which consists of 272K annotated real life images. Ideally, the dataset should consist of input-output pairs (such as the example on the right in Figure 1) which are 'solvable' as opposed to the one on the left which is ambiguous. However, as we explore the original *Who's Waldo* dataset, we encounter a great portion of cases that resemble the left example in Figure 1, unsolvable data with insufficient contextual cues. Given such context, if we do not recognize who exactly is in the picture, even we human beings cannot tell which name is who. We can then only make predictions with biased assumptions, such as the first named person would always be on the leftmost, or the main subject would always make up the largest area. Such biases in the original dataset may explain why the heuristic methods perform very strongly, outperforming random guessing by a big 27% increase in test accuracy and trailing the top benchmark only by 6%. We believe a fair dataset should not encourage approaches to adopt biases to such an extent, and thus the original baseline model overestimates its performance.

Inspired by dataset debiasing works such as VQA-CP (Agrawal et al., 2018) and GQA-OOD (Kervadec et al., 2021), we create a debiased collection of 84K annotated image-captions out of the *Who's Waldo* dataset by filtering out all biased data with insufficient context. We evaluate the quality of our new dataset by applying the original heuristic methods as well as *Who's Waldo*'s benchmark model. Results show that our debiased dataset greatly reduces the heuristic biases from the original dataset and provides the PCVG task a more practical baseline for future developments.

2 Related Work

Dataset Debiasing. We take many inspirations from previous studies on uncurated datasets. A task dataset if not curated properly could lead to methods that cheat their ways through without learning generalized information. For example, VQAv2 (Goyal et al., 2017) addresses the imbalance between language and images in VQAv1 (Antol et al., 2015) which results in visual information being ignored and inflated model performance. VQA-CP (Agrawal et al., 2018) and GQA-OOD (Kervadec et al., 2021) were designed to test model performance if spurious correlations exist in the training dataset. Cadene et al. (2019); Chen et al. (2020a); Gokhale et al. (2020) are bias-aware techniques that mitigate dataset bias with modeling and data augmentation. Ye and Kovashka (2021) introduce exploits by matching repeated texts in questions and answers to achieve high scores in Visual Commonsense Reasoning (Zellers et al., 2019).

We also learn from various techniques to amend priors, biases, or shortcuts in datasets. REPAIR (Li and Vasconcelos, 2019) uses resampling to fix representation biases in image datasets. Dasgupta et al. (2018) incorporate compositional information into sentence embeddings for Natural Language Inference. DQI (Mishra et al., 2020) offers quantitative metrics to assess biases in automated dataset creation in Natural Language Processing. Le Bras et al. (2020) introduce adversarial measures to mitigate biases in various Natural Language Processing and Computer Vision tasks.

Visual Grounding. The PCVG task adapts previous supervised Visual Grounding models as its original baselines. The Visual Grounding task is defined as locating specific objects in an image from a textual description. First established by Karpathy et al. (2014), following researches have evolved into extracting attention information such as works by Deng et al. (2018) and Endo et al. (2017). A huge variation of datasets for Visual Grounding have also been created, including Flicker30k (Plummer et al., 2015), Visual Genome (Krishna et al., 2017), and RefCOCO (Yu et al., 2016).

Referring Expression Comprehension (REC). An active branch from Visual Grounding, the Referring Expression Comprehension task (Rohrbach et al., 2016) is no longer restricted to object categories. Instead its goal is to relate a free region in an image to a sentence description. Mattnet (Yu et al., 2018) is one prominent approach that leverages both attention features and relation extraction for the objects in the image. Qiao et al. (2020) offers a comprehensive survey on this topic.

Human Detection. A specialized category under Object Detection, detecting humans with bounding boxes in images nowadays can easily use open source toolboxes including MMDetection (Chen

Selection	**Train**	**Val.**	**Test**	**Unused**
Original	179073	6740	6741	79193
no-verb	125585	3446	3529	34366
conjunct-names	16446	2237	2227	15693
Ours	45884	2102	2049	33611

Table 1: The data in our debiased dataset are filtered and regrouped from all four splits in the original. Notice, examples such as the **Left** in Figure 1 can have both zero verb and at least three conjunct names.

et al., 2019) or Detectron (Wu et al., 2019) that are trained on large-scale real life image datasets like COCO (Lin et al., 2014). Recent works such as DarkPose (Zhang et al., 2020) also attempt to utilize human pose information to better single out human traits from complex background.

3 Method

In this section, we introduce the Person-centric Visual Grounding task, discuss the original *Who's Waldo* dataset, and provide our analysis of shortcuts, biases, and other issues that we discovered in the dataset. We describe the process via which we curate, debias, and filter the dataset.

3.1 The Task

The Person-centric Visual Grounding task is defined as follows. The givens are an image **I**, a set of m $\geq$ 1 person detections **B** (in form of bounding boxes), and a corresponding image caption **T** where its tokens contain references to n $\geq$ 1 persons. For each referred person, we look for the best matching detection from the givens. We also assume no two persons can be matched with the same detection.

3.2 The *Who's Waldo* Dataset

The dataset consists of 272K real-life captioned images sourced from the free Wikimedia Commons repository. Each image pictures individuals under the 'People by name' category on Wikimedia Commons, while its caption describes the scene and explicitly mentions the featured people in real names. Key dataset creation procedures, text pre-processing, identifying person entities in captions, detecting bounding boxes of people in images, and generating ground truths linking bounding boxes and names, are all done with existing automated tools such as FLAIR (Akbik et al., 2019) and MMDetection (Chen et al., 2019). To prevent misuse, in the publicly released version, all the real names in the captions are replaced with the [NAME] token, but references between bounding boxes and token indices are given in individual annotation files. This is equivalent to masking each name with indexed placeholders such as PERSON1, PERSON2, etc. Amongst the entirety of 272K annotated samples, 179K samples are used for training, 6.7K for validation, and 6.7K for testing. Each test sample is supposed to either *mention at least two persons* or *choose from at least two bounding boxes*. The original test set is further validated manually on Amazon Mechanical Turk.

3.3 Biases in *Who's Waldo*

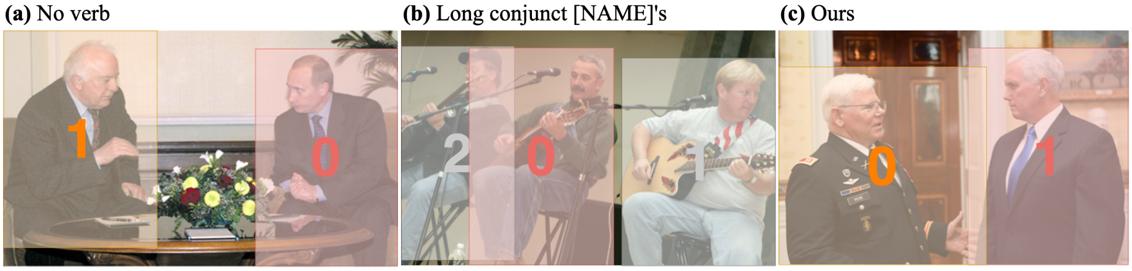
The premise of the Person-centric Visual Grounding task is to use ONLY the caption text and the image as the cues to find out the correct bounding box from the image per mentioned name. However, we observe a large portion of the original *Who's Waldo* dataset does not provide sufficient contexts and can only be solved by heuristic methods. We discuss two major types of biases that we discover in the following sections.

The first type no-verb is that the caption text contains zero detectable verbs. Since linguistically a verb is the crucial part of an action that assigns participants with semantic roles, we technically have no way to tell who performs or who receives an action without verbs. For example in Figure 2(a), we are unable to tell who is who from the image and the no-verb caption alone, unless we recognize Vladimir Putin or the Georgian President with external knowledge.

The second type conjunct-names is that the caption contains a long chain of conjunct referred names. Shown in Figure 2(b), all the referred names share the verb *perform*, joined together only with conjunct words such as *and* or *along with*. With no indication of the order amongst these persons, we can only resort to a naive positional order such as left-to-right. But since we may also have extra bounding boxes as choices, such naive assumption is indeed unreliable. Figure 2(b) is such an example that the first mentioned name is not always the one in the left-most bounding box.

3.4 Data Curation for De-biasing

In order to resolve the aforementioned limitations of the original dataset, we utilize two pipelines in SpaCy ver 3.0 (Honnibal et al., 2020) to filter out the biased data. We apply the POS-Tagging pipeline to find out if sentences in an image cap-



Caption: President [NAME] with Georgian President [NAME].

GT: [NAME] → 0 , [NAME] → 1

Caption: Country artist [NAME] along with musicians [NAME] and [NAME] *perform* for troops during a visit to Camp Fallujah, Iraq Nov. 26, 2005.

GT: [NAME] → 0

Caption: Vice President [NAME] *speaks* to Retired U.S. Army Capt. [NAME] before the Medal of Honor ceremony in Washington, D.C., Oct. 23, 2017. [NAME] was *awarded* the Medal of Honor during the Vietnam War ...

GT: [NAME] → 1, [NAME] → 0

Figure 2: **(a)** and **(b)** represent the two major types of insufficient and biased data that we filter out. **(c)** represents the ones we choose for our debiased dataset. We label all detected verbs in *italic*. We apply color coding to indicate different person entities in a caption. We also use gray bounding boxes to refer to those 'incorrect options' not included in ground truth, such that in the ground truth of **(b)**, the only pair we need to associate is [NAME] with Bounding Box 0, while the two other bounding boxes serve as mere distractions.

tion contain verbs in any form of conjugation. In parallel, we use the Dependency Parsing pipeline to examine if any [NAME] token conjuncts with more than one [NAME]'s from different referred persons. We jointly filter out any example that either (a) contains zero verbs, or (b) has at least three conjunct referred person names in a sentence. For both pipelines, we replace the [NAME] tokens that refer to the same person in a caption with a random popular first name, so that the natural language-based SpaCy pipelines can yield more accurate results. Both pipelines use the state-of-the-art `en-web-core-trf` model which is built on RoBERTa (Liu et al., 2019).

Ultimately, our filtering procedure produces 84K qualifying image-caption pairs. Table 1 shows the distribution of samples sourced from each split of the original through our two debiasing pipelines. We utilize data from the unused yet legitimately annotated 79K samples of the original dataset. We reorganize and split all the qualifying 84K samples into 74K for training, 5K for validation, and 5K for test. Our new test set does not overlap with the original training set. Similarly to the design of the original, we enforce that all samples in our new test set involves no trivial case that contains exactly one referred name and exactly one bounding box. We also make sure that any test set sample always has at least one name-to-bounding-box pair as ground truth.

4 Experiments and Baselines

Setup. We evaluate the quality of our debiased dataset with the same heuristic and Transformer-based methods from the original paper. We also train the benchmark model on both the original and our new training set. We report the accuracies obtained from our new test set as the new baselines.

Heuristics. We inherit the original heuristic measures to study the potential biases of our debiased dataset versus those of the original dataset. Alongside Random guessing, we assign the names in the caption to the bounding boxes sorted by: (a) decreasing area size (Big → Small), (b) left-to-right upper-left coordinates (L → R (All)), and (c) left-to-right upper-left coordinates of the largest d bounding boxes, d being the larger between the number of bounding boxes and the number of names in a test case (L → R (Largest)).

Transformer-based Models. We adapt the original benchmark *Who's Waldo* model to our debiased dataset and see how well it can perform under the updated contexts. The benchmark model is a multi-layer multi-modal Transformer (Vaswani et al., 2017). Based on UNITER (Chen et al., 2020b), it learns to maximize the similarities between the corresponding person names and bounding boxes while minimize the similarities between those that do not match up. We fine-tune the *Who's Waldo* model with pre-trained weights from UNITER.

Analysis of Results. Table 2 shows the test set accuracies for the original dataset and our debi-

Method	Training Set	Test Set	Test Accuracy	Δ_r	Δ_h
Random	–	Original Test	30.9	0.0	–
Big → Small	–	Original Test	48.2	+17.3	–
L → R (All)	–	Original Test	38.4	+7.5	–
L → R (Largest)	–	Original Test	57.7	+26.8	0.0
Gupta et al.	COCO	Original Test	39.3	+8.4	-18.4
SL-CCRF	Flickr30K Entities	Original Test	46.4	+15.9	-11.3
MAttNet	RefCOCOg	Original Test	44.0	+13.1	-13.7
Who's Waldo	Original Train	Original Test	63.5	+32.6	+5.8
Random	–	Our Test	31.0	0.0	–
Big → Small	–	Our Test	43.8	+12.8	–
L → R (All)	–	Our Test	32.4	+1.4	–
L → R (Largest)	–	Our Test	44.3	**+13.3**	0.0
Who's Waldo	Original Train	Our Test	50.2	+19.2	+5.9
Who's Waldo	Our Train	Our Test	**54.0**	+23.0	**+9.7**
Who's Waldo	Our Train	*Biased samples* of Original Test	48.2	–	–

Table 2: Evaluation on the test sets using the original *What's Waldo* and our debiased dataset. Δ_r denotes relative improvement over random guessing, and Δ_h denotes relative improvement over the best heuristic. The *biased samples* represents a total of 4.7K samples from the original test set that are filtered out by our debiasing procedure. The original work also compares its baseline performance with multiple pre-trained visual grounding models, such as Gupta et al. (2020) trained with COCO (Lin et al., 2014), SL-CCRF (Liu and Hockenmaier, 2019) trained with Flickr30K Entities (Plummer et al., 2015), and MAttNet (Yu et al., 2018) trained with RefCOCOg (Mao et al., 2016). All reported accuracies in this table are the strongest averaged performances per setting and fall within a fluctuation of $\pm 1\%$.

ased dataset. We find that the heuristic measures have overall lower performance on our new dataset, meaning we have successfully reduced the effects of the positional and the size-based biases from the original dataset. Most significantly, we have lowered L → R (All) from +7.5% to +1.4%, almost equal to randomness. Even the strongest L → R (Largest) heuristic has been lowered from +26.8% all the way down to +13.3% as well. Our dataset is thus proven less biased compared to the original.

We also show that our dataset has better practicality for the task. Measured with our new test set, the performance of the *Who's Waldo* benchmark model trained with the original training set performs 3.8% lower than that trained with our new, smaller training set. Meanwhile, the test accuracy gap between the Transformer-based method and the heuristic methods has become larger using our debiased dataset, widened from 5.8% to 9.7%. In addition, using the filtered *biased samples* from the original test set on our new trained model yields an even lower performance at 48.2%, which indicates our new baseline model now adopts fewer biases during training compared to the original. Altogether with the lowered new baseline accuracy of 54.0%, we argue that our debiased dataset improves the quality of contextual cues that supervised models can learn from, and leaves more applicable room for improvements in the future.

5 Conclusion

We present a refined dataset for the PCVG task with samples that contain contextual information required for the task. We address prominent biases that we identified in the original task dataset by filtering out a large number of unsolvable cases, and report new baseline performances on the new benchmark. Our refined dataset can serve as a more reliable benchmark to enable fair comparisons for new modeling techniques and training protocols.

Acknowledgements

This research was supported by grants from DARPA SAIL-ON, DARPA KAIROS, NSF 1816039, and NSF 2132724. The views and opinions of the authors expressed herein do not necessarily state or reflect those of the funding agencies and employers.

Ethical Considerations

Our curated dataset is available at `https://github.com/fpsluozi/tofindwaldo` . We will also follow the same licensing and data sharing policy as the original Who's Waldo dataset.

References

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. 2018. Don't just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4971–4980.

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Remi Cadene, Corentin Dancette, Matthieu Cord, Devi Parikh, et al. 2019. Rubi: Reducing unimodal biases for visual question answering. *Advances in neural information processing systems*, 32.

Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. 2019. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*.

Long Chen, Xin Yan, Jun Xiao, Hanwang Zhang, Shiliang Pu, and Yueting Zhuang. 2020a. Counterfactual samples synthesizing for robust visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10800–10809.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020b. Uniter: Universal image-text representation learning. pages 104–120.

Yuqing Cui, Apoorv Khandelwal, Yoav Artzi, Noah Snavely, and Hadar Averbuch-Elor. 2021. Who's waldo? linking people across text and images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1374–1384.

Ishita Dasgupta, Demi Guo, Andreas Stuhlmüller, Samuel J Gershman, and Noah D Goodman. 2018. Evaluating compositionality in sentence embeddings. *arXiv preprint arXiv:1802.04302*.

Chaorui Deng, Qi Wu, Qingyao Wu, Fuyuan Hu, Fan Lyu, and Mingkui Tan. 2018. Visual grounding via accumulated attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7746–7755.

Ko Endo, Masaki Aono, Eric Nichols, and Kotaro Funakoshi. 2017. An attention-based regression model for grounding textual phrases in images. In *IJCAI*, pages 3995–4001.

Zhiyuan Fang, Tejas Gokhale, Pratyay Banerjee, Chitta Baral, and Yezhou Yang. 2020. Video2commonsense: Generating commonsense descriptions to enrich video captioning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 840–860.

Tejas Gokhale, Pratyay Banerjee, Chitta Baral, and Yezhou Yang. 2020. Mutant: A training paradigm for out-of-distribution generalization in visual question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 878–892.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6904–6913.

Tanmay Gupta, Arash Vahdat, Gal Chechik, Xiaodong Yang, Jan Kautz, and Derek Hoiem. 2020. Contrastive learning for weakly supervised phrase grounding. In *European Conference on Computer Vision*, pages 752–768. Springer.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spacy: Industrial-strength natural language processing in python.

Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. 2014. Deep fragment embeddings for bidirectional image sentence mapping. *Advances in neural information processing systems*, 27.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 787–798.

Corentin Kervadec, Grigory Antipov, Moez Baccouche, and Christian Wolf. 2021. Roses are red, violets are blue... but should vqa expect them to? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2776–2785.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. In *International Conference on Machine Learning*, pages 1078–1088. PMLR.

Yi Li and Nuno Vasconcelos. 2019. Repair: Removing representation bias by dataset resampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9572–9581.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Jiacheng Liu and Julia Hockenmaier. 2019. Phrase grounding by soft-label chain conditional random field. pages 5115–5125.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11–20.

Swaroop Mishra, Anjana Arunkumar, Bhavdeep Sachdeva, Chris Bryan, and Chitta Baral. 2020. Dqi: Measuring data quality in nlp. *arXiv preprint arXiv:2005.00816*.

Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649.

Yanyuan Qiao, Chaorui Deng, and Qi Wu. 2020. Referring expression comprehension: A survey of methods and datasets. *IEEE Transactions on Multimedia*.

Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. 2016. Grounding of textual phrases in images by reconstruction. In *European Conference on Computer Vision*, pages 817–834. Springer.

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

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. 2019. Detectron2. `https://github.com/facebookresearch/detectron2`.

Keren Ye and Adriana Kovashka. 2021. A case study of the shortcut effects in visual commonsense reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3181–3189.

Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. 2018. Mattnet: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1307–1315.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. 2016. Modeling context in referring expressions. In *European Conference on Computer Vision*, pages 69–85. Springer.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6720–6731.

Feng Zhang, Xiatian Zhu, Hanbin Dai, Mao Ye, and Ce Zhu. 2020. Distribution-aware coordinate representation for human pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7093–7102.

Translate-Train Embracing Translationese Artifacts

Sicheng Yu[♠] **Qianru Sun**[♠] **Hao Zhang**[♣◇] **Jing Jiang**[♠]
[♠]Singapore Management University, Singapore
[♣]Nanyang Technological University, Singapore
[◇]Centre for Frontier AI Research, A*STAR, Singapore
`scyu.2018@phdcs.smu.edu.sg, hzhang26@outlook.com`
`{qianrusun,jingjiang}@smu.edu.sg`

Abstract

Translate-train is a general training approach to multilingual tasks. The key idea is to use the translator of the target language to generate training data to mitigate the gap between the source and target languages. However, its performance is often hampered by the artifacts in the translated texts (translationese). We discover that such artifacts have common patterns in different languages and can be modeled by deep learning, and subsequently propose an approach to conduct translate-train using *T*ranslationese *E*mbracing the effect of *A*rtifacts (TEA). TEA learns to mitigate such effect on the training data of a source language (whose original and translationese are both available), and applies the learned module to facilitate the inference on the target language. Extensive experiments on the multilingual QA dataset TyDiQA demonstrate that TEA outperforms strong baselines.

1 Introduction

Cross-lingual transfer has drawn wide attention in recent years (Hu et al., 2020; Liang et al., 2020). It has great potentials to be applied in advanced industries and real applications such as for improving dialog and advertisement systems in multilingual countries (Schuster et al., 2019; Yu et al., 2021). It aims to reuse NLP models trained on a *source* language for the task of a *target* language. The most intuitive method is transfer learning by leveraging pre-trained multilingual language models (LMs) such as mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020). These pre-trained LMs encode different languages into a joint space of multilingual representations (Wu and Dredze, 2019; Lauscher et al., 2020), and they perform well especially for zero-shot cross-lingual tasks (Wu and Dredze, 2019; Lauscher et al., 2020). Another method orthogonal to this is called translate-train (Hu et al., 2020; Fang et al., 2021). It translates training data from the source language into the target language and uses the translated texts for training. Our paper focuses on this second method.

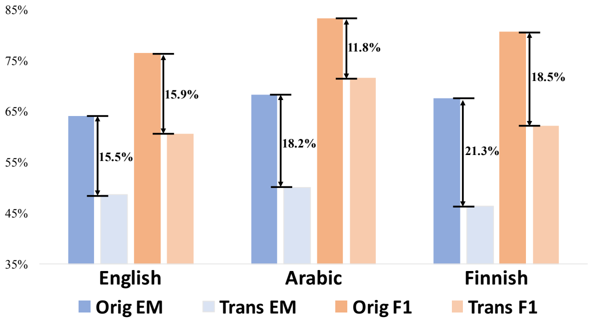


Figure 1: Monolingual QA performance comparison between (i) training using originals and (ii) training using translated texts on TyDiQA. Note that for each language, training data and test data are in the same language. "EM" stands for Exact Match.

Translate-train mitigates the language gap between the source and the target languages in multilingual tasks in a straightforward manner as it directly generates the needed target language training samples. However, the translation process introduces artifacts in the translated texts (*i.e.*, translationese[1]). It has been observed that translationese often exhibits features such as stylistic ones that are different from text written directly in the same language (which we call the *originals*) and thus can mislead model training (Selinker, 1972; Volansky et al., 2015; Bizzoni et al., 2020). In Figure 1, we show that even in a monolingual setting where training and test data are in the same language, when the test data are original texts, using translationese to train a QA model results in substantial performance drop compared with using originals for training.

To tackle the issue with translationese artifacts, inspired by domain mapping techniques (Zhu et al., 2017), we explore the idea of projecting originals

[1] We refer to texts directly written by humans in a certain language as *originals* of that language and translated texts (translated by either humans or machines) as *translationese*.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 362 - 370
May 22-27, 2022 ©2022 Association for Computational Linguistics

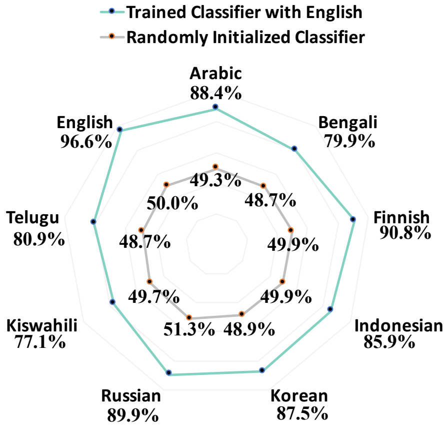


Figure 2: The XLM-R (Conneau et al., 2020) classification results of translationese for different languages on TyDiQA (Clark et al., 2020). We use the classifiers trained with only English pairs (originals and translationese), or randomly initialized (without any training). More details of the experiments are given in Appendix B.

and translationese into a common embedding space to close their gap. Since only the originals of the source language are available under the translate-train setting, whether this idea is feasible depends on whether the projection function is learnable from the source language and transferable to other languages. Therefore, we first conduct experiments to investigate if translationese artifacts, or patterns of differences between orginals and translationese, are recognizable and transferable across languages by deep learning models. Specifically, we train a binary classifier to distinguish English originals from English translationese. We then test the effectiveness of this binary classifier on other languages. Our intuition is that (1) if the model converges, it suggests that the patterns of translationese artifacts can be potentially learned to some extent, and 2) if the trained model recognizes the translationese of other languages, it means the model can likely transfer the learned patterns across different languages. Our results in Figure 2 validate both: 1) The model converges well and achieves 97% accuracy on English, the training language. (2) It also performs reasonably well on other languages (77% ~ 91%).

Based on the above intuition and validation, we propose a Translationese Embracing Artifacts (TEA) method that projects originals and translationese into a common space to mitigate the translationese artifacts. TEA explicitly learns a mapping function from originals to translationese using originals and translationese of the source language (English in our experiments), where learning is through minimizing the distance between the mapped representation of originals and of the corresponding translationese. TEA then applies this mapping function to the originals of the target language during the testing stage. For evaluation, we conduct experiments on multilingual QA using the TyDiQA dataset (Clark et al., 2020)[2]. Our results show that TEA outperforms translate-train baselines and SOTA translationese mitigation methods designed for machine translation (Marie et al., 2020; Wang et al., 2021).

2 Related Work

The effect of translationese has been widely studied in translation tasks (Lembersky et al., 2012; Zhang and Toral, 2019; Edunov et al., 2020; Graham et al., 2020; Freitag et al., 2020). Some works focus on mitigating or controlling the effect of translationese, *e.g.*, tagged training (Marie et al., 2020; Riley et al., 2020; Wang et al., 2021), which are adopted as baselines in our paper. In the field of cross-lingual transfer, there are very few works about translationese. Artetxe et al. (2020) is the only attempt for translate-test and zero-shot learning. In contrast, we focus on translate-train and aim to mitigate the artifacts in translationese.

Our research is also related to domain adaptation (DA) that aims to transfer the knowledge from a source domain to target domains. Our original-to-translationese projection function can be seen as something similar to projecting source domain and target domain data into a common space, which has been used before for domain adaptation (Zhu et al., 2017; Shen et al., 2017).

3 Our Approach (TEA)

Let $\mathbf{x}$ represent the input text and $\mathbf{y}$ represent the output label. $\mathcal{X}$ denotes the domain (*i.e.*, all possible values) of $\mathbf{x}$ and $\mathcal{Y}$ is the set of labels. The input $\mathbf{x}$ comes from different languages, and it can be either originals or translationese during training. Specifically, we use $\mathcal{X}_{\text{src, orig}}$ to denote the domain of *source* language *originals*, and define $\mathcal{X}_{\text{trgt, orig}}$ and $\mathcal{X}_{\text{trgt, trans}}$ in a similar way (where $_{\text{trgt}}$ refers to the target language and $_{\text{trans}}$ refers to translationese). We further use back-translation (Sennrich et al., 2016) to generate *source* language *trans-*

[2] To the best of our knowledge, TyDiQA is the only large-scale multilingual benchmark dataset where test data is original text written by humans.

lationese (*i.e.*, the source language originals are first translated into a pivot language and then translated back into the source language), denoted as $\mathcal{X}_{\text{src, trans}}$, for the purpose of learning a mapping function to project originals and translationese into the same space.

We now present our TEA method. Our ultimate goal is to learn a mapping function $f : \mathcal{X}_{\text{trgt, orig}} \to \mathcal{Y}$, which takes target language originals as input. However, we only have $\mathcal{D}_{\text{src, orig}} \in \mathcal{X}_{\text{src, orig}} \times \mathcal{Y}$ and $\mathcal{D}_{\text{trgt, trans}} \in \mathcal{X}_{\text{trgt, trans}} \times \mathcal{Y}$ during training. The challenge is that an f learned from either $\mathcal{D}_{\text{src, orig}}$ or $\mathcal{D}_{\text{trgt, trans}}$ may not work effectively on $\mathcal{X}_{\text{trgt, orig}}$ because of the differences between the source and the target languages and between originals and translationese. To mitigate the differences between the source and the target languages, we rely on pre-trained multilingual language models, as many existing works do. As for the differences between originals and translationese, based on the idea discussed in Section 1, we propose to mitigate the translationese artifacts of the target language using an original-to-translationese mapping function, and because of the lack of target originals, we propose to learn the original-to-translationese mapping function from the *source* language.

To concretely illustrate our idea, we break down the mapping from $\mathcal{X}$ to $\mathcal{Y}$ into the following steps[3]:

Multilingual Projection (MP): First, input $\mathbf{x}$ is projected into a language-agnostic multilingual space by using a pre-trained multilingual LM. We use $\mathcal{X}_{\text{ml}}$ to denote the projected multilingual space, and f_{MP} is a multilingual projection (*i.e.*, the multilingual LM) that maps an input $\mathbf{x}$ in any language into $\mathcal{X}_{\text{ml}}$.

Original-to-Translationese Projection (OTP): Suppose $\mathcal{X}_{\text{ml}}$ consists of two subspaces: $\mathcal{X}_{\text{ml}} = \mathcal{X}_{\text{ml, orig}} \bigcup \mathcal{X}_{\text{ml, trans}}$, where $\mathcal{X}_{\text{ml, orig}}$ and $\mathcal{X}_{\text{ml, trans}}$ denote the multilingual representations of any originals and translationese, respectively. To close the gap between originals and translationese, we define an original-to-translationese projection function $f_{\text{OTP}} : \mathcal{X}_{\text{ml, orig}} \to \mathcal{X}_{\text{ml, trans}}$ to convert the vector representation of a piece of originals to its corresponding representation of translationese.

Language-Agnostic QA (QA): The last step is a language-agnostic classifier for the QA task itself. We use $f_{\text{QA}} : \mathcal{X}_{\text{ml, trans}} \to \mathcal{Y}$ to denote it.

Given an input $\mathbf{x}$, depending on whether it is from originals or translationese, we use different compositions of the functions above to map $\mathbf{x}$ to $\mathbf{y}$:

$$\mathbf{y} = \begin{cases} f_{\text{QA}} \circ f_{\text{OTP}} \circ f_{\text{MP}}(\mathbf{x}) & \mathbf{x} \in \mathcal{X}_{*,\text{ orig}}, \\ f_{\text{QA}} \circ f_{\text{MP}}(\mathbf{x}) & \mathbf{x} \in \mathcal{X}_{*,\text{ trans}}. \end{cases}$$

Here $\circ$ represents the composition of two functions, *i.e.*, $f \circ g(x) = f(g(x))$, and $*$ denotes source language or target languages. More concretely, for $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}_{\text{src, orig}}$, we use $\mathcal{X}_{\text{src, orig}} \xrightarrow{f_{\text{MP}}} \mathcal{X}_{\text{ml, orig}} \xrightarrow{f_{\text{OTP}}} \mathcal{X}_{\text{ml, trans}} \xrightarrow{f_{\text{QA}}} \mathcal{Y}$; for $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}_{\text{trgt, trans}}$, we use $\mathcal{X}_{\text{trgt, trans}} \xrightarrow{f_{\text{MP}}} \mathcal{X}_{\text{ml, trans}} \xrightarrow{f_{\text{QA}}} \mathcal{Y}$.

As discussed in Section 1, we make use of the source language originals and translationese to learn f_{OTP}. Specifically, for $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}_{\text{src, orig}}$, we use $\mathbf{x}' \in \mathcal{X}_{\text{src, trans}}$ to represent its corresponding translationese, *i.e.*, generated by back-translation (Sennrich et al., 2016) through a pivot language. Let $\{(\mathbf{x}, \mathbf{x}')\} \in \mathcal{D}_{\text{src, pairs}}$ denotes all the pairs of originals and translationese in the source language. Then, we minimize the distance between $f_{\text{OTP}}(f_{\text{MP}}(\mathbf{x}))$ and $f_{\text{MP}}(\mathbf{x}')$ to optimize f_{OTP}.

In summary, the loss function consists of the following three components:

$$\begin{aligned} L = & \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}_{\text{src, orig}}} l(f_{\text{QA}} \circ f_{\text{OTP}} \circ f_{\text{MP}}(\mathbf{x}), \mathbf{y}) \\ & + \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}_{\text{trgt, trans}}} l(f_{\text{QA}} \circ f_{\text{MP}}(\mathbf{x}), \mathbf{y}) \\ & + \sum_{(\mathbf{x},\mathbf{x}') \in \mathcal{D}_{\text{src, pairs}}} (1 - g(\mathbf{x}, \mathbf{x}')), \end{aligned}$$

where $g(\mathbf{x}, \mathbf{x}') = \cos(f_{\text{OTP}}(f_{\text{MP}}(\mathbf{x})), f_{\text{MP}}(\mathbf{x}')$. $l(\cdot, \cdot)$ is standard cross entropy loss and $\cos(\cdot, \cdot)$ is the cosine similarity function.

Model Details. For f_{MP}, we use XLM-R (Conneau et al., 2020). For f_{OTP}, we utilize a transformer layer (Vaswani et al., 2017). f_{QA} is implemented by a linear layer.

4 Experiments

Dataset. We conduct experiments on TyDiQA (Clark et al., 2020). Specifically, we evaluate our approach on the gold-passage subtask of TyDiQA, which includes 9 languages. We set English as source language and others as target languages, and report the results on target languages. During training, we utilize translated training data in *all* target languages for joint training. We use Exact Match (EM) and F1 scores as evaluation metrics.

Implementation. Translations of English training data for target languages are from XTREME (Hu

[3] A diagram showing the pipeline is in Appendix A.

Method	D	ar	bn	fi	id	ko	ru	sw	te	*med*	*all-in-one*	*avg*
STT	✘	40.4/67.6	47.8/64.0	53.2/70.5	61.9/77.4	10.9/31.9	42.1/67.0	48.1/66.1	43.6/70.1	45.7/67.3	45.2/67.2	43.5/64.3
FILTER	✘	50.8/72.8	56.6/70.5	57.2/73.3	59.8/76.8	12.3/33.1	46.6/68.9	65.7/77.4	50.4/69.9	53.7/71.7	51.6/70.3	49.9/67.8
STT*	✘	58.0/76.6	54.6/70.2	59.0/74.8	64.7/80.2	48.0/61.6	49.5/71.2	58.7/74.6	57.0/76.2	57.5/74.7	56.8/74.4	56.2/73.2
TAG*	✔	56.9/76.4	55.5/70.0	59.4/75.2	64.4/79.6	48.6/61.7	49.1/70.4	60.7/76.0	57.8/76.4	57.4/75.5	56.9/74.5	56.5/73.2
TST*	✔	58.4/75.5	60.2/72.2	58.3/74.4	65.5/78.9	49.3/62.6	49.0/69.7	63.5/76.7	56.2/76.1	58.3/75.0	57.3/74.1	57.6/73.3
GRL*	✔	57.6/75.6	58.4/72.6	59.7/74.8	65.3/79.9	49.6/62.2	49.1/70.4	62.9/76.9	58.2/77.0	58.3/75.2	57.6/74.6	57.6/73.7
TEA*	✔	56.5/76.1	60.2/74.9	60.9/76.5	63.6/79.3	48.6/61.4	51.5/72.0	66.7/78.9	60.7/78.7	**60.5/76.3**	**58.6/75.6**	**58.6/74.7**

Table 1: Main results (Exact Match / F1 scores) on TyDiQA. All methods use XLM-R as backbone. The "D" column indicates whether the model design considers translationese artifacts. The columns "ar" to "te" are different target languages. The "med" and "avg" columns denote median and average performance across the 8 target languages. The "all-in-one" column is the result by combining all data as one dataset. * indicates our implementation.

et al., 2020) and translationese English is translated by Google Cloud Translation. German (de) is selected as the default pivot language in back-translation. More details are in the Appendix B.

Baselines. We compare our model with the following baselines: (1) Standard Translate-Train (STT) (Devlin et al., 2019). (2) FILTER (Fang et al., 2021) is an advanced translate-train method fully utilizing the parallel data. (3) Tagging (TAG) (Marie et al., 2020), which distinguishes originals and translationese by adding a tag for machine translation. (4) Two-Stage Training (TST) (Wang et al., 2021), which is another approach to address the gap between translationese and originals for machine translation. It first uses the combination of them for training followed by another round of training only on originals. (5) Gradient Reversal Layer (GRL) (Ganin and Lempitsky, 2015), which is a general DA method.

Main results. Table 1 summarizes the comparison between our TEA and the baselines. We make the following observations: (1) TEA outperforms all baselines. For instance, TEA surpasses STT by 2.4% (EM) and 1.5% (F1) on average, which demonstrates the effectiveness of our method. (2) Methods considering translationese artifacts generally perform better than methods without such design, which reinforces the importance of mitigating translationese artifacts. (3) Compared to the baselines for translationese artifacts, TEA still shows its superiority. We highlight that our OTP module with explicit projection is better than implicit DA approaches. E.g., TAG only uses different tags to distinguish the translationese from originals. (4) The improvements from TEA across different languages are different. For high-resource[4] target languages, TEA brings more gains on the languages in Indo-European family, *e.g.*, ru, and marginal gains on others, *e.g.*, ar. For low-resource target languages, the performance improvements are obvious, *e.g.*, sw. It is because both language model and machine translation model are of lower quality on low-resource languages, and thus mitigating the gap between translationese and originals shows more effectiveness in such scenario. For high-resource languages, TEA prefers Indo-European languages, which are closer to English.

Settings	EM	F1
STT	56.2	73.2
(1) STT+$\mathcal{X}_{\text{src, trans}}$	56.6	73.2
(2) STT+params	56.3	73.5
(3) TOP	57.9	74.1
(4) MLP in OTP	56.7	73.3
(5) MSE loss	58.0	73.9
Full method	**58.6**	**74.7**

Table 2: Ablation study on TyDiQA. We report the average EM and F1 performance on the 8 target languages.

Ablation studies. We conduct in-depth ablation studies to analyze TEA. Specifically, we explore the following settings: (1) Since we use 11% more data in TEA (unlabeled $\mathcal{X}_{\text{src, trans}}$) compared to STT, here we add labeled $\mathcal{X}_{\text{src, trans}}$ in STT. (2) Since we use additional 0.38% parameters (OTP) in our method compared to STT, here we add the same OTP module in STT. (3) We replace the Original-to-Translationese Projection (OTP) by Translationese-to-Original Projection (TOP). (4) We replace the self-attention layer in OTP with a multi-layer perceptron (MLP). (5) We replace the cosine distance function in loss with mean square function. The results are summarized in Table 2. Compared to the variants, our full method performs best over all settings. (1)/(2) incorporate additional data/parameters, which demonstrates the improvement of our method is not caused by the two factors.

[4] Here we distinguish high-resource and low-resource according to XLM-R (Conneau et al., 2020).

Settings	Language Family	EM	F1
Scottish (gd)	Indo-European	58.8	74.0
Korean (ko)	Koreanic	57.8	74.0
Chinese (zh)	Sino-Tibetan	57.6	73.8
German (de)	Indo-European	58.6	74.7

Table 3: Experiment results of utilizing different language as pivot language for generating $\mathcal{X}_{\text{src, trans}}$.

(3) proves that TOP still mitigates the artifacts, but OTP obtaining better performance. We argue that it is because most of the training data is translationese. (4) and (5) demonstrate the effectiveness of our loss function and architecture.

Pivot Languages Analysis. Here we study the effect of pivot language used in generating $\mathcal{X}_{\text{src, trans}}$. Specifically, we select four pivot languages, *i.e.*, German (de), Scottish (gd), Korean (ko) and Chinese (zh), for evaluation. We fix our approach and only replace the $\mathcal{X}_{\text{src, trans}}$ used in OTP. The results are reported in Table 3. We observe that pivot languages from Indo-European family are superior to that from other language families. We believe it is because the training data of other target languages in translate-train is translated from English, while English belongs to the Indo-European family.

5 Conclusions

We aim to mitigate the translationese artifacts when training translate-train models. After varifying the transferability of the translationese patterns across languages, we propose the TEA that mitigates artifacts using a source language and to facilitate the prediction on unseen target languages. Our approach is simple and generic. Moreover, our results on multilingual QA show its effectiveness.

Ethical Considerations

Although our method requires fine-tuning of the pre-trained multilingual language model, the computational cost of our experiments is not high. We utilize two pieces of NVIDIA V100 and it takes around 1 hour for the fine-tuning process. This is partly due to the relatively small QA training dataset used for fine-tuning. It is possible that if our method is applied to either a much larger training dataset for fine-tuning or a much larger pre-trained language model, the computational cost and power consumption will go up. To reduce such costs, one way is to fine-tune only part of the pre-trained language model. Another way is to apply the recently proposed Adapter method (Houlsby et al., 2019) to fine-tune the language model.

Our method relies on machine translation systems. It has been found in a previous study that industrial MT systems as well as SOTA academic MT systems may suffer from gender bias (Stanovsky et al., 2019), and it would not be surprising if other types of societal biases and stereotypes are also found in machine translated texts. If our method uses translationese containing societal biases, our learned original-to-translationese projection function will likely also contain such biases, which may affect the fairness of the final trained system. However, this is not due to our method but rather the translated text we use. Nevertheless, this is something we need to keep in mind if our method is adopted for real applications.

Acknowledgments

This research is supported by the National Research Foundation, Singapore under its Strategic Capabilities Research Centres Funding Initiative, and partially supported by the Agency for Science, Technology and Research (A*STAR) under its AME YIRG Grant (Project No. A20E6c0101) and the A*STAR Career Development Fund (Project No. C210812033). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2020. Translation artifacts in cross-lingual transfer learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7674–7684.

Yuri Bizzoni, Tom S Juzek, Cristina Espana-Bonet, Koel Dutta Chowdhury, Josef van Genabith, and Elke Teich. 2020. How human is machine translationese? comparing human and machine translations of text and speech. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 280–290.

Jonathan H Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

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

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

Sergey Edunov, Myle Ott, Marc'Aurelio Ranzato, and Michael Auli. 2020. On the evaluation of machine translation systems trained with back-translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2836–2846.

Yuwei Fang, Shuohang Wang, Zhe Gan, Siqi Sun, and Jingjing Liu. 2021. Filter: An enhanced fusion method for cross-lingual language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 14, pages 12776–12784.

Markus Freitag, David Grangier, and Isaac Caswell. 2020. Bleu might be guilty but references are not innocent. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 61–71.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR.

Yvette Graham, Barry Haddow, and Philipp Koehn. 2020. Translationese in machine translation evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 72–81.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multitask benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499.

Gennadi Lembersky, Noam Ordan, and Shuly Wintner. 2012. Adapting translation models to translationese improves smt. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 255–265.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, et al. 2020. Xglue: A new benchmark datasetfor cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018.

Fuli Luo, Wei Wang, Jiahao Liu, Yijia Liu, Bin Bi, Songfang Huang, Fei Huang, and Luo Si. 2021. VECO: Variable and flexible cross-lingual pre-training for language understanding and generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 3980–3994.

Benjamin Marie, Raphael Rubino, and Atsushi Fujita. 2020. Tagged back-translation revisited: Why does it really work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5990–5997.

Parker Riley, Isaac Caswell, Markus Freitag, and David Grangier. 2020. Translationese as a language in "multilingual" nmt. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7737–7746.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805.

Larry Selinker. 1972. Interlanguage. *International Review of Applied Linguistics in Language Teaching*, pages 209–231.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. *Advances in Neural Information Processing Systems*, 30.

Gabriel Stanovsky, Noah A Smith, and Luke Zettlemoyer. 2019. Evaluating gender bias in machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1679–1684.

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

Vered Volansky, Noam Ordan, and Shuly Wintner. 2015. On the features of translationese. *Digital Scholarship in the Humanities*, 30(1):98–118.

Shuo Wang, Zhaopeng Tu, Zhixing Tan, Shuming Shi, Maosong Sun, and Yang Liu. 2021. On the language coverage bias for neural machine translation. In *Findings of the 59th Annual Meeting of Association for Computational Linguistics*.

Xiangpeng Wei, Rongxiang Weng, Yue Hu, Luxi Xing, Heng Yu, and Weihua Luo. 2020. On learning universal representations across languages. In *International Conference on Learning Representations*.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844.

Puxuan Yu, Hongliang Fei, and Ping Li. 2021. Cross-lingual language model pretraining for retrieval. In *Proceedings of the Web Conference 2021*, pages 1029–1039.

Mike Zhang and Antonio Toral. 2019. The effect of translationese in machine translation test sets. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 73–81.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232.

A Training Pipeline

Figure 3 illustrates the training pipelines our method. The goal is to map the originals and translationese domains into the same embedding space for prediction. Specifically, the module on the top of the figure is to train the $\mathcal{D}_{\text{src, orig}}$ and $\mathcal{D}_{\text{trgt, trans}}$, *i.e.*, the first two terms of loss function, while the module at the bottom aims to map the originals-translationese pairs in $\mathcal{D}_{\text{src, pairs}}$ into same space, *i.e.*, the last term of loss function.

B Implementation

General Implementation. We adopt the HuggingFace Transformers (Wolf et al., 2020) toolkit to implement the pre-trained language model, *i.e.*, XLM-R. The maximal input length, *i.e.*, concatenation of question and passage tokens, is set as 384. We also utilize a document sliding window with stride length of 128 to tackle the long passage issue. The learning rate and batch size are set as $2e-5$ and 32, respectively. We use back-translate (Sennrich et al., 2016) to generate the translationese. Back-translate means to translate the source language to a pivot language and then translate back to the source language. By doing this, we are able to obtain the translationese of source language.

Implementation of Experiment in Figure 1. TyDiQA (Clark et al., 2020) provides both training and testing datasets of originals for all languages. Here we adopt the originals training data to generate the corresponding translationese training data through back-translation [5]. The results in Figure 1 are obtained by training originals and generated translationese data for en, ar and fi, respectively. Note all the translationese is generated by the Google Cloud Translation[6] service, where the English translationese is generated by back-translationese with de as pivot language, the translationeses of ar and fi use en as pivot language. The The test set is originals of each language.

Implementation of Experiment in Figure 2. Similarly, we generate the translationese of the originals for each language using Google Cloud Translation service, where en is set as pivot language for non-English languages, and German for English language. We split the originals-translationese pairs of English into two groups, where 80% samples are used for training, and the rest 20% samples together with all pairs of other languages are used for evaluation. As the originals and translationese are paired, a random guess could achieve 50% accuracy for all languages ideally.

Implementation of TEA. It is worth noting that we can only access the originals of English and the translationese of other target languages during training. We use the translationese data, *i.e.*, the target language data translated from English, from

[5] We emphasize that non-English originals data is only utilized in Figure 1 and Figure 2 for analysis purpose. In addition, we only utilize the originals data of English in experiment, which follows the same settings as previous works, for translate-train.

[6] https://cloud.google.com/translate

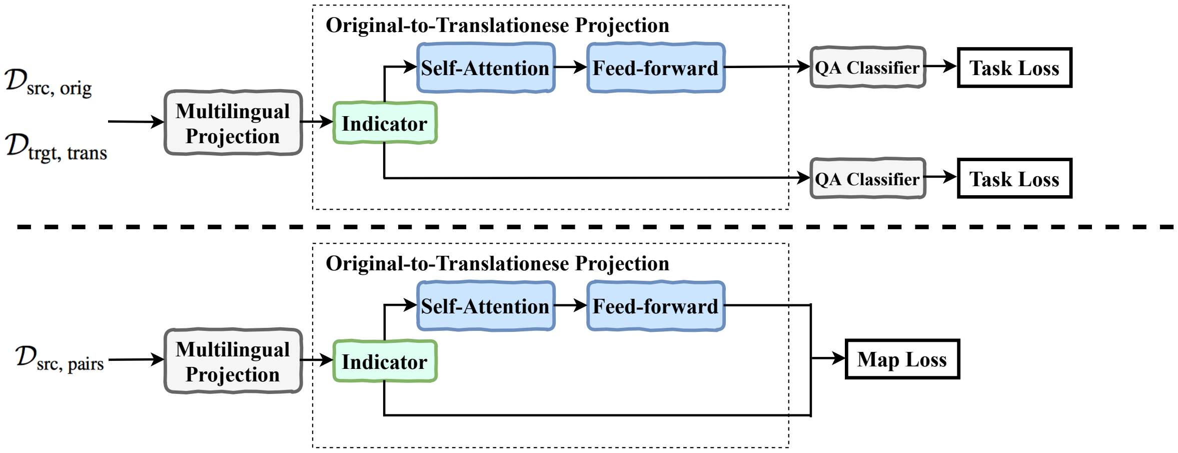

Original-to-Translationese Projection
$\mathcal{D}_{\text{src, orig}}$
$\mathcal{D}_{\text{trgt, trans}}$
Multilingual Projection
Indicator
Self-Attention
Feed-forward
QA Classifier
Task Loss
QA Classifier
Task Loss
Original-to-Translationese Projection
$\mathcal{D}_{\text{src, pairs}}$
Multilingual Projection
Indicator
Self-Attention
Feed-forward
Map Loss

Figure 3: Overall training pipeline of TEA. All modules are shared. Indicator is used to forward different kind of data into upper path or lower path, *i.e.*, originals data for upper path and translationese data for lower path. The task losses are standard cross entropy loss and the map loss is computed by cosine distance function.

XTREME official website[7], while the translated data from XTREME is utilized in translate-train for all previous works (Fang et al., 2021). Besides, we also augment translationese of English, which is generated by back-translation, in our TEA. Again, we resort to the Google Cloud Translation service to generate the translationese for all experiments in Section 4, where the German is set as pivot language by default.

C Data and Parameters

The sample sizes of the data sets in all 9 languages are equal, since they are all translated from English originals training data. The standard translate-train (STT) directly adopt the data samples of 9 languages for training. In addition to the data samples of 9 languages, we also incorporate the English translationese, leading to 11% more samples used compared to SST. Besides, our Original-to-Translationese Projection (OTP) module also introduce additional parameters compared to SST.

D Additional Experiments

Main Results. In this part, we replenish the TyDiQA results of two advanced multilingual language models, *i.e.*, VECO (Luo et al., 2021) and HICTL (Wei et al., 2020) in Table 4.

Originals-Translationese Pair Sample. In Figure 4, we list examples of originals-translationese pair in English used for TEA training.

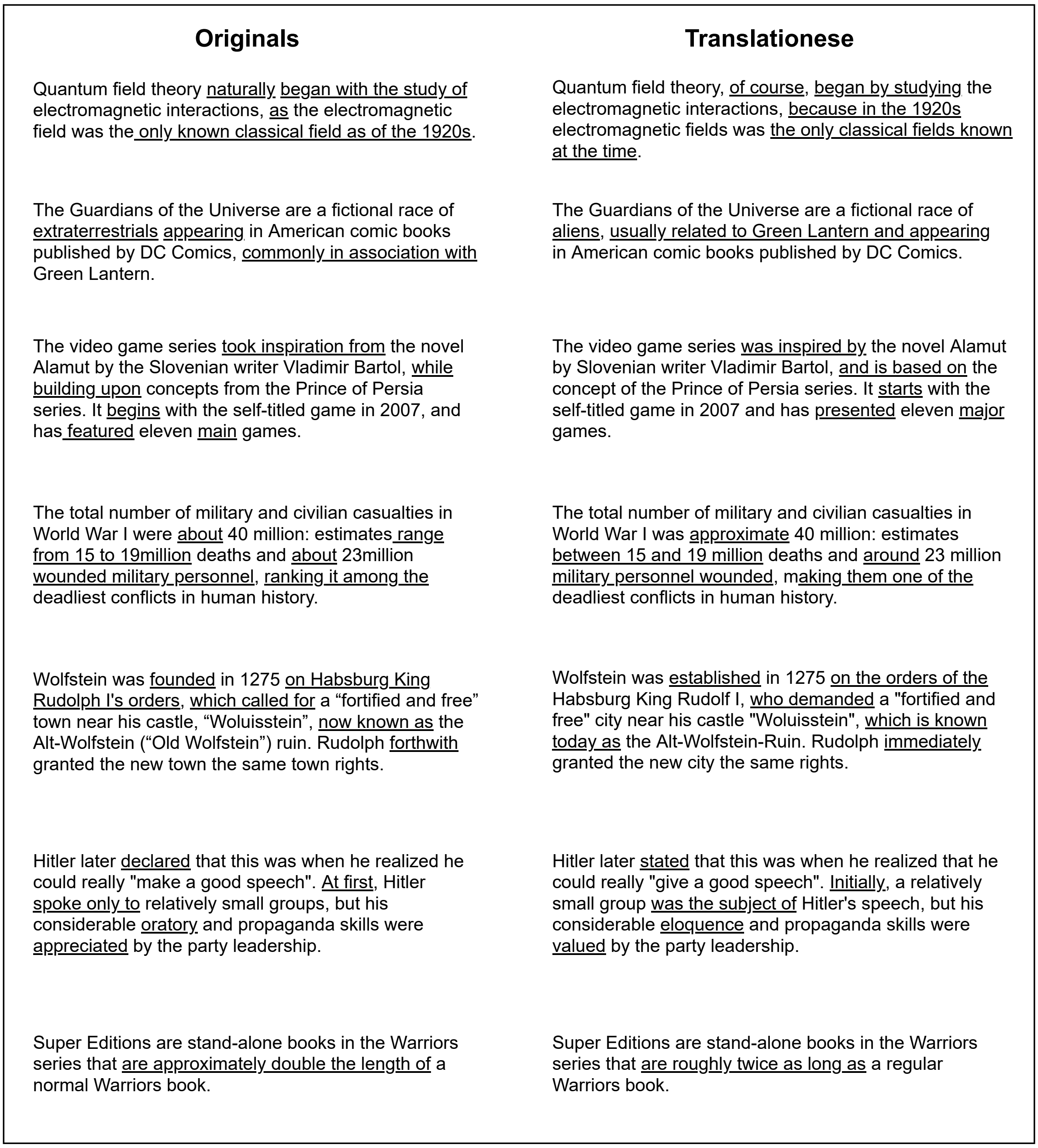
Effect of Translation Quality on Translationese English. Here we conduct an ablation study about the effect of translation quality on translationese English used in cosine distance loss. Due to the limited resource, we are unable to train a machine translation model from scratch by ourselves. Instead, we select the free Google Translate toolkit[8] (compared to the paid Google Cloud service) as the proxy of low-quality translator. We fix all the implementation settings and change the translationese English data only. Consequently, we obtain the average performance of $EM/F1$ = 58.0/73.9. The result indicates that a better translator is more effective for the translationese English generation. It is because that the low-quality translator may create more translation errors, then those errors are propagated during training, which hinders the learning of the originals to translationese mapping.

[7] https://console.cloud.google.com/storage/browser/xtreme_translations

[8] https://translate.google.com

LM	Method	ar	bn	fi	id	ko	ru	sw	te	avg
XLM-R	STT	40.4/67.6	47.8/64.0	53.2/70.5	61.9/77.4	10.9/31.9	42.1/67.0	48.1/66.1	43.6/70.1	43.5/64.3
	FILTER	50.8/72.8	56.6/70.5	57.2/73.3	59.8/76.8	12.3/33.1	46.6/68.9	65.7/77.4	50.4/69.9	49.9/67.8
	STT*	58.0/76.6	54.6/70.2	59.0/74.8	64.7/80.2	48.0/61.6	49.5/ 71.2	58.7/74.6	57.0/76.2	56.2/73.2
	TAG*	56.9/76.4	55.5/70.0	59.4/75.2	64.4/79.6	48.6/61.7	49.1/70.4	60.7/76.0	57.8/76.4	56.5/73.2
	TST.*	58.4/75.5	60.2/72.2	58.3/74.4	65.5/78.9	49.3/62.6	49.0/69.7	63.5/76.7	56.2/76.1	57.6/73.3
	GRL*	57.6/75.6	58.4/72.6	59.7/74.8	65.3/79.9	49.6/62.2	49.1/70.4	62.9/76.9	58.2/77.0	57.6/73.7
	TEA*	56.5/76.1	60.2/74.9	60.9/76.5	63.6/79.3	48.6/61.4	51.5/72.0	66.7/78.9	60.7/78.7	**58.6/74.7**
HICTL	STT	52.1/72.7	45.3/64.6	61.8/79.1	61.7/79.6	37.1/53.8	51.6/71.3	56.9/71.5	51.7/68.3	52.3/70.1
VECO	STT	57.5/77.0	56.6/72.2	59.3/76.6	64.4/80.0	52.2/63.4	50.5/72.8	67.1/79.4	58.0/76.0	58.2/74.7

Table 4: Main results on TyDiQA dataset. "LM": language models; "avg" denotes average performance across 8 languages; "∗": our implementation.



Originals	Translationese
Quantum field theory naturally began with the study of electromagnetic interactions, as the electromagnetic field was the only known classical field as of the 1920s.	Quantum field theory, of course, began by studying the electromagnetic interactions, because in the 1920s electromagnetic fields was the only classical fields known at the time.
The Guardians of the Universe are a fictional race of extraterrestrials appearing in American comic books published by DC Comics, commonly in association with Green Lantern.	The Guardians of the Universe are a fictional race of aliens, usually related to Green Lantern and appearing in American comic books published by DC Comics.
The video game series took inspiration from the novel Alamut by the Slovenian writer Vladimir Bartol, while building upon concepts from the Prince of Persia series. It begins with the self-titled game in 2007, and has featured eleven main games.	The video game series was inspired by the novel Alamut by Slovenian writer Vladimir Bartol, and is based on the concept of the Prince of Persia series. It starts with the self-titled game in 2007 and has presented eleven major games.
The total number of military and civilian casualties in World War I were about 40 million: estimates range from 15 to 19million deaths and about 23million wounded military personnel, ranking it among the deadliest conflicts in human history.	The total number of military and civilian casualties in World War I was approximate 40 million: estimates between 15 and 19 million deaths and around 23 million military personnel wounded, making them one of the deadliest conflicts in human history.
Wolfstein was founded in 1275 on Habsburg King Rudolph I's orders, which called for a "fortified and free" town near his castle, "Woluisstein", now known as the Alt-Wolfstein ("Old Wolfstein") ruin. Rudolph forthwith granted the new town the same town rights.	Wolfstein was established in 1275 on the orders of the Habsburg King Rudolf I, who demanded a "fortified and free" city near his castle "Woluisstein", which is known today as the Alt-Wolfstein-Ruin. Rudolph immediately granted the new city the same rights.
Hitler later declared that this was when he realized he could really "make a good speech". At first, Hitler spoke only to relatively small groups, but his considerable oratory and propaganda skills were appreciated by the party leadership.	Hitler later stated that this was when he realized that he could really "give a good speech". Initially, a relatively small group was the subject of Hitler's speech, but his considerable eloquence and propaganda skills were valued by the party leadership.
Super Editions are stand-alone books in the Warriors series that are approximately double the length of a normal Warriors book.	Super Editions are stand-alone books in the Warriors series that are roughly twice as long as a regular Warriors book.

Figure 4: Examples of originals-translationese pair in English from TyDiQA (Clark et al., 2020). The main differences are underlined.

C-MORE: Pretraining to Answer Open-Domain Questions by Consulting Millions of References

Xiang Yue[1,*], Xiaoman Pan[2], Wenlin Yao[2], Dian Yu[2], Dong Yu[2], and Jianshu Chen[2]

[1]The Ohio State University
[2]Tencent AI Lab
yue.149@osu.edu
{xiaomanpan,wenlinyao,yudian,dyu,jianshuchen}@tencent.com

Abstract

We consider the problem of pretraining a two-stage open-domain question answering (QA) system (retriever + reader) with strong transfer capabilities. The key challenge is how to construct a large amount of high-quality question-answer-context triplets without task-specific annotations. Specifically, the triplets should align well with downstream tasks by: (i) covering a wide range of domains (for open-domain applications), (ii) linking a question to its semantically relevant context with supporting evidence (for training the retriever), and (iii) identifying the correct answer in the context (for training the reader). Previous pretraining approaches generally fall short of one or more of these requirements. In this work, we automatically construct a large-scale corpus that meets all three criteria by consulting millions of references cited within Wikipedia. The well-aligned pretraining signals benefit both the retriever and the reader significantly. Our pretrained retriever leads to 2%-10% absolute gains in top-20 accuracy. And with our pretrained reader, the entire system improves by up to 4% in exact match.[1]

1 Introduction

Open-domain question answering (QA) aims to extract the answer to a question from a large set of passages. A simple yet powerful approach adopts a two-stage framework (Chen et al., 2017; Karpukhin et al., 2020), which first employs a retriever to fetch a small subset of relevant passages from large corpora (i.e., *retriever*) and then feeds them into a *reader* to extract an answer (text span) from them. Due to its simplicity, a sparse retriever such as TF-IDF/BM25 is generally used together with a trainable reader (Min et al., 2019). However, recent advances show that transformer-based dense retrievers trained on supervised data (Karpukhin et al., 2020) can greatly boost the performance, which better captures the semantic relevance between the question and the correct passages. Such approaches, albeit promising, are restricted by the limited amount of human annotated training data.

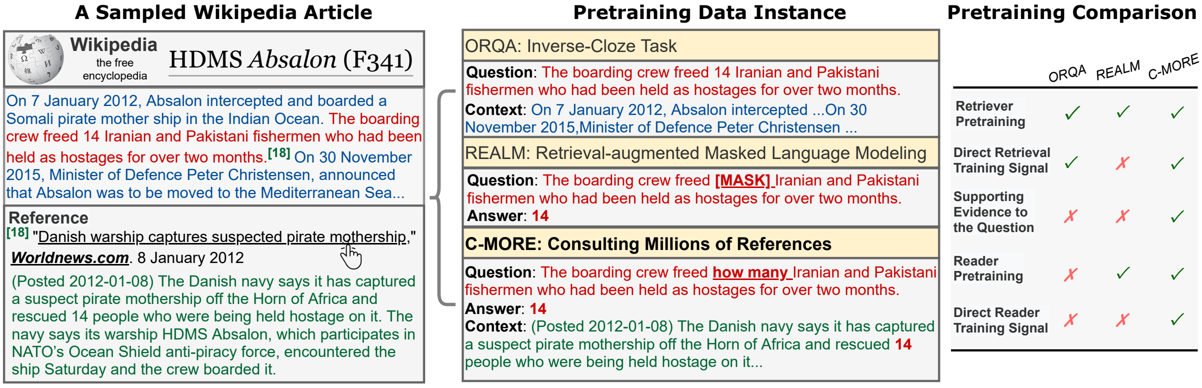
Inspired by the recent progresses of language models pretraining (Devlin et al., 2019; Lee et al., 2019; Guu et al., 2020; Sachan et al., 2021), we would like to address the following central question: *can we pretrain a two-stage open-domain QA system (retriever + reader) without task-specific human annotations?* Unlike general language models, pretraining such a system that has strong transfer capabilities to downstream open-domain QA tasks is challenging. This is mainly due to the lack of well-aligned pretraining supervision signals. In particular, we need the constructed pretraining dataset (in the form of question-answer-context triplets) to: (i) cover a wide range of domains (for open-domain applications), (ii) link a question to its semantically relevant context with supporting evidence (for training the retriever), and (iii) identify the correct answer in the context (for training the reader).

There have been several recent attempts in addressing these challenges. ORQA (Lee et al., 2019) creates pseudo query-passage pairs by randomly sampling a sentence from a paragraph and treating the sampled sentence as the question while the rest sentences as the context. REALM (Guu et al., 2020) adopts a retrieve-then-predict approach, where the context is dynamically retrieved during training and an encoder (reader) predicts

[*] Work was done when interning at Tencent AI Lab.

[1] Our code, data, and pretrained models are available at: https://github.com/xiangyue9607/C-MORE

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 371 - 377
May 22-27, 2022 ©2022 Association for Computational Linguistics



On 7 January 2012, Absalon intercepted and boarded a Somali pirate mother ship in the Indian Ocean. The boarding crew freed 14 Iranian and Pakistani fishermen who had been held as hostages for over two months.[18] On 30 November 2015, Minister of Defence Peter Christensen, announced that Absalon was to be moved to the Mediterranean Sea...

Reference

[18] "Danish warship captures suspected pirate mothership," *Worldnews.com*. 8 January 2012

(Posted 2012-01-08) The Danish navy says it has captured a suspected pirate mothership off the Horn of Africa and rescued 14 people who were being held hostage on it. The navy says its warship HDMS Absalon, which participates in NATO's Ocean Shield anti-piracy force, encountered the ship Saturday and the crew boarded it.

ORQA: Inverse-Cloze Task

Question: The boarding crew freed 14 Iranian and Pakistani fishermen who had been held as hostages for over two months.
Context: On 7 January 2012, Absalon intercepted ...On 30 November 2015,Minister of Defence Peter Christensen ...

REALM: Retrieval-augmented Masked Language Modeling

Question: The boarding crew freed **[MASK]** Iranian and Pakistani fishermen who had been held as hostages for over two months.
Answer: 14

C-MORE: Consulting Millions of References

Question: The boarding crew freed **how many** Iranian and Pakistani fishermen who had been held as hostages for over two months.
Answer: **14**
Context: (Posted 2012-01-08) The Danish navy says it has captured a suspected pirate mothership off the Horn of Africa and rescued **14** people who were being held hostage on it...

	ORQA	REALM	C-MORE
Retriever Pretraining	✓	✓	✓
Direct Retrieval Training Signal	✓	✗	✓
Supporting Evidence to the Question	✗	✗	✓
Reader Pretraining	✗	✓	✓
Direct Reader Training Signal	✗	✗	✓

Figure 1: Different pretraining methods for open-domain QA. Our **C-MORE** pretrains both retriever and reader by using direct signals extracted from millions of references cited in the verified knowledge source.

the masked token in the question based on the retrieved context. The retriever pretraining signals constructed in these approaches are not aligned with question-context pairs in open-domain QA settings. For example, as shown in Figure 1, the context (in blue color) of ORQA pretraining data instance does not contain direct supporting evidence to the question. Likewise, the dynamically retrieved context in REALM cannot be guaranteed to contain direct supporting evidence either. In addition, existing pretraining methods (Lee et al., 2019; Guu et al., 2020) mostly focus on the retriever and do not jointly provide direct pretraining signals for the reader (Figure 1).

To meet all three aforementioned criteria, we propose a pretraining approach named **C**onsulting **M**illions **O**f **RE**ferences (**C-MORE**), which automatically constructs pretraining data with well-aligned supervision signals (Figure 1). Specifically, we first extract three million statement-reference pairs from Wikipedia along with its cited references. Then, we transform them into question-answer-context triplets by replacing a potential answer span in the statement (e.g., *"14"* in the Figure 1) by an interrogative phrase (e.g, *"how many"*). Such kind of pseudo triplets are in the exact same form as human-annotated ones, and the question is linked to the context that contains the most direct-supporting evidence, a highly desirable feature for open-domain QA tasks. We experiment the pretraining with a widely-adopted open-domain QA system, Dense Passage Retriever (DPR) (Karpukhin et al., 2020). The experimental results show that our pretrained retriever not only outperforms both sparse and dense retrieval baselines in the zero-shot retrieval setting (2%-10% absolute gain in top-20 accuracy), but also leads to further improvement in the downstream task finetuning. By integrating with our pretrained reader, the entire open-domain pretraining improves the end-to-end QA performance by 4% in exact match.

2 Method

Recall that we want to automatically construct a large-scale open-domain QA pretraining dataset that satisfies three criteria: (i) The dataset should cover a wide range of domains for the open-domain QA purpose. (ii) The context passage is semantically relevant to the question and contains direct supporting evidence for answering the question. (iii) The correct answer span in the context passage for answering the question should also be identified for training the reader. This section first discusses how to extract a large amount of statement-reference pairs from the Wikipedia and then explain how to construct pseudo question-answer-context triplets for pretraining open-domain QA systems.

2.1 Statement-Reference Pairs Collection

Wikipedia articles usually contain a list of knowledge sources (references) at the end that are verified by human editors to support the statements in the articles (Li et al., 2020). And the reference documents always consist of strong supporting evidence to the statements. For example, as shown in Figure 1, the document (in green color) contains the direct evidence *"...rescued 14 people who were being held hostage on it..."* to support the query (red text) *"The boarding crew freed 14 Iranian and Pakistani fishermen who had been held as hostages over two months"*. Additionally, such knowledge sources are often organized in a good structure and can be automatically extracted and processed. Moreover, the statement-reference pairs in Wikipedia cover

Data Type	Dataset	Train	Dev	Test
Pretraining	C-MORE	2.96M	40K	-
Finetuning QA Data	NaturalQuestion	58,880	8,757	3,610
	TriviaQA	60,413	8,837	11,313
	WebQuestion	2,474	361	2,032

Table 1: Statistics of pretraining and finetuning data.

a wide range of topics and domains. Thus, when converted into question-context pairs, they satisfy the first two criteria and are suitable for training an accurate dense retriever at a large scale.

In our study, we extract around six million statement-reference pairs from Wikipedia. We filter the pairs whose reference documents are not reachable and finally obtain around three million statement-reference pairs (see statistics in Appendix Table 1). The data collection method we proposed is very general and therefore can be easily extended to other domains, e.g., WikiEM (wikem.org) for medical domain or other languages, e.g., Baidu Baike (baike.baidu.com) for Chinese.

2.2 QAC Triplets Construction

We now explain how to further convert the statement-reference pairs into question-answer-context pairs. Inspired by previous unsupervised extractive QA work (Lewis et al., 2019), we extract entities as potential answers to construct pseudo question-answer-context pairs where an answer span is extracted from the context given an question to accommodate the extractive QA setting. Specifically, we first adopt an off-the-shelf named entity recognition tool spaCy (Honnibal and Montani, 2017) to identify entities in each query. Next, we filter the entities that do not appear in the evidence based on string matching. If multiple entities are found, we sample one of them as the potential answer to the query. The sampled entity in the query is replaced by an interrogative phrase based on the entity type (e.g., a [DATE] entity will be replaced by phrases such as *"when"*, *"what date"*. In this way, we can construct question-answer-context triplets to train open-domain QA models. See more question reformation rules in Appendix Table 5).

3 Experiment

3.1 Experimental Setup

Pretraining Model Architecture. Since conceptually the construed triplets is in the same format as the annotated QA data, they can be used to pretrain any existing neural open-domain QA model. Here, we adopt DPR (Karpukhin et al., 2020), which consists of a dual-encoder as the retriever and a BERT reader, considering its effectiveness and popularity. Specifically, the retriever first retrieves top-k (up to 400 in our experiment) passages, and the reader assigns a passage score to each retrieved passage and extracts an answer with a span score. The span with the highest passage selection score is regarded as the final answer. The reader and retriever can be instantiated with different models and we use BERT-base-uncased for both of them following (Karpukhin et al., 2020).

Pretraining Data Processing. For our extracted pseudo question-answer-context triplets, sometimes the context (reference document) is too long to fit into a standard BERT (maximum 512 tokens) in the DPR model. Thus, we chunk a long document into n-word text blocks with a stride of m. Without loss of generality, we use multiple combinations of n and m: $n = \{128, 256, 512\}$, $n = \{64, 128, 256\}$. Then we calculate relevance scores (using BM25) of the derived blocks with the question and select the most relevant block as the context. Note that the retrieval step is done within the single document (usually less than 20 text blocks). In contrast, the baseline model (Section 3.2) - sparse retriever BM25 - looks up the entire knowledge corpus (20M text blocks). In this way, we can automatically collect the most relevant context that supports the query from a long article.

Finetuning QA Datasets. We consider three popular open-domain QA datasets for finetuning: NaturalQuestions (NQ) (Kwiatkowski et al., 2019), TriviaQA (TQA) (Joshi et al., 2017), and WebQuestions (WebQ) (Berant et al., 2013), whose statistics are shown in Table 1.

Following the setting of DPR (Karpukhin et al., 2020), we use the Wikipedia as the knowledge source and split Wikipedia articles into 100-word units for retrieval. All the datasets we use are the processed versions from the DPR implementation.

Overlap between Pretraining and Finetuning Datasets. Though both C-MORE and downstream QA data are constructed based on Wikipedia, the overlap between them would be very little. C-MORE extracts queries from Wikipedia while the queries of downstream QA data are annotated by human. C-MORE extracts contexts from the external referenced pages (general Web) while the downstream QA data extract contexts from Wikipedia.

Implementation Details. For pretraining, we set

Settings	Methods	Training Data	Top-20 Accuracy			Top-100 Accuracy		
			NQ	TQA	WebQ	NQ	TQA	WebQ
Unsupervised	BM25	-	59.1*	66.9*	55.0*	73.7*	76.7*	71.1*
	ORQA (Lee et al., 2019)	Wikipedia	50.6†	57.5†	-	66.8†	73.6†	-
	REALM (Guu et al., 2020)	Wikipedia	59.8†	68.2†	-	74.9†	79.4†	-
	C-MORE	**Wikipedia**	**61.9**	**72.2**	**62.7**	**75.8**	**81.3**	**78.5**
Domain Aaptation	DPR-NQ	NaturalQuestion	-	69.7	69.0	-	79.2	78.8
	+ **C-MORE**	+ Wikipedia	-	**72.8**	**71.2**	-	**81.6**	**81.3**
	DPR-TQA	TriviaQA	69.2	-	71.5	80.3	-	81.0
	+ **C-MORE**	+ Wikipedia	**71.0**	-	**74.3**	**81.7**	-	**83.2**
	DPR-WebQ	WebQ	56.1	66.1	-	70.7	77.6	-
	+ **C-MORE**	+ Wikipedia	**67.3**	**74.2**	-	**79.2**	**82.6**	-
Supervised	DPR-supervised	Supervised Data	78.4*	79.4*	73.2*	85.4*	85.0*	81.4*
	+ **C-MORE**	+ Wikipedia	**80.3**	**81.3**	**75.0**	**86.7**	**85.9**	**83.2**

Table 2: Overall retrieval performance of different models. Results marked with "*" are from DPR (Karpukhin et al., 2020), "†" are from (Sachan et al., 2021) and "-" means it does not apply to the current setting.

Row	Model Architecture	Retriever		Reader		NQ	TQA	WebQ
		Pretrain	Finetune	Pretrain	Finetune			
1	ORQA (Lee et al., 2019)	✓	✓	✗	✓	33.3	45.0	36.4
2	REALM (Guu et al., 2020)	✓	✓	✓	✓	40.4	-	40.7
3	DPR (Karpukhin et al., 2020)	✓	✗	✓	✗	11.3	24.8	4.5
4		✗	✗	✗	✓	32.6	52.4	29.9
5		✓	✗	✗	✓	**35.3**	**55.1**	**32.1**
6		✗	✓	✗	✓	41.5	56.8	34.6
7		✓	✓	✗	✓	**41.9**	58.6	35.6
8		✓	✓	✓	✓	41.6	**60.3**	**38.6**

Table 3: End-to-end QA performance based on different retrievers and readers. Note that we only test the effectiveness of **C-MORE** based on the DPR (Karpukhin et al., 2020) model architecture. ORQA and REALM are listed here as references. The retriever of Row 4 is BM25, which does not involve either pretraining or finetuning.

training epochs to 3, batch size to 56 for retrievers and 16 for readers, and learning rate to 2e-5. We select the best checkpoint based on the pretraining dev set. For finetuning, we use the same set of hyperparameters as the original DPR paper. The comparing baselines ORQA (Lee et al., 2019) and REALM (Guu et al., 2020) use 288-token truncation over Wikipedia, which are not directly comparable to our results. To enable a fair comparison, we report the retrieval results from a recent paper (Sachan et al., 2021), which uses the same retrieval corpus as ours.

3.2 Retrieval Performance

We consider three settings to demonstrate the usefulness of our pretrained retriever.

Unsupervised. We assume no annotated training QA pairs are available. In this setting, We compare our method with existing unsupervised retrievers: a sparse retriever BM25 and two pretrained dense retrievers ORQA and REALM.

Domain Adaptation. We consider the condition in which there are QA training pairs in the source domain but no training data in the target domain. The task is to obtain good retrieval performance on the target test set only using source training data. We compare our method with two baselines: one is to directly train a dense retriever on the source domain while the other is to first pretrain a dense retriever on our constructed corpus and then finetune it on the source domain training set.

Supervised. In this setting, all the annotated QA training instances are used. Similar to the previous setting, we compare a supervised retriever with and without our **C-MORE** pretraining.

For all settings, we report the top-k retrieval accuracy ($k \in \{20, 100\}$) on the test set following (Karpukhin et al., 2020). See the overall retrieval performance of different models in each setting in Table 2. We have the following observations.

In the **unsupervised** setting, compared with the strong sparse retrieval baseline BM25, our pretrained dense retriever shows significant improvement. For example, we obtain around 7% absolute improvement in terms of both Top-20 and Top-100 accuracy on the WebQuestion dataset. Compared with pretrained dense retrievers (i.e., ORQA and REALM), our pretrained model outperforms them by a large margin. This is not surprising as our pretraining data contain better aligned retrieval supervision signals: reference documents often have supporting evidence for the question while their retrieval training signals are relatively indirect.

In the **domain adaptation** and **supervised** settings, our pretrained dense retriever provides a better finetuning initialization and leads to improvement compared with randomly initialized DPR models. Another surprising result is that our pretrained dense retriever even outperforms some DPR domain adaptation models. For example, on the TriviaQA testing set, our pretrained DPR model achieves 72.2% top-20 and 81.3% top-100 accuracy while the DPR-NQ model obtains 69.7% and 79.2% respectively. This indicates that our pretrained dense retriever can generalize well even without using any annotated QA instances.

All the results demonstrate the usefulness and generalization of our pretrained dense retriever for open-domain QA tasks.

3.3 End-to-End QA performance

We now examine how our pretrained retriever and reader improve the end-to-end QA performance, measured in exact match (EM). The results are shown in Table 3, from which we make the following observations. (i) Surprisingly, our fully-unsupervised system (pretrained retriever + pretrained reader) shows a certain level of open-domain QA ability (see row #3). For example, on TriviaQA, our fully-unsupervised system can answer around 25% of questions correctly. (ii) Compared to the system with BM25 retriever (row #4), the one with our pretrained dense retriever (line #5) retrieves more relevant passages, leading to better QA performance. (iii) Initializing either the retriever or the reader from our pretrained checkpoint can lead to further improvement (rows #6-#8). For example, on the TriviaQA and WebQuestion datasets, our entire pipeline pretrain leads to about 4% absolute gain in terms of EM. Note that on the WebQuestion dataset, all the DPR models perform worse than REALM, this is because of the limited training data of WebQuestion. The issue can be easily solved by adding *Multi* datasets for finetuning according to (Karpukhin et al., 2020).

3.4 Computational Resource Comparison

In addition to the performance gain, another benefit of **C-MORE** is its training scalability. We compare the **C-MORE** pretraining with ORQA and REALM in terms of computational resources they use in Table 4. As can be seen, **C-MORE** only requires reasonable GPU computational resources, which could be normally conducted on an academic-level computational platform. On the contrary, due to the lack of direct retrieval supervision, ORQA and REALM often needs more computational resources and requires more training steps to converge.

	GPU			TPU		
	#cards	batch size	Train steps	#cards	batch size	Train steps
ORQA	128	4096	100K	-	4096	100K
REALM	240	-	100K	64	512	200K
C-MORE	8	56	20K	-	-	-

Table 4: Computational resource comparison between different retriever pretraining methods. Our **C-MORE** provides more direct retrieval pretraining signals, thus leading to fast converge. ORQA and REALM GPU setups are from (Sachan et al., 2021) and TPU setups are from their original papers.

4 Conclusion

This paper proposes an effective approach for pretraining open-domain QA systems. Specifically, we automatically construct three million pseudo question-answer-context triplets from Wikipedia that align well with open-domain QA tasks. Extensive experiments show that pretraining a widely-used open-domain QA model (DPR) on our constructed data achieves promising performance gain in both retrieval and QA accuracies. Future work includes exploring the effectiveness of the constructed data on more open-domain QA models (e.g., REALM) and training strategies (e.g., joint optimizing the retriever and reader).

Acknowledgements

The authors would thank the anonymous reviewers for their insightful comments and suggestions. The authors would also thank the colleagues in Tencent AI Lab for their internal discussions and feedback.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT'19*, pages 4171–4186. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910.

Zhongli Li, Wenhui Wang, Li Dong, Furu Wei, and Ke Xu. 2020. Harvesting and refining question-answer pairs for unsupervised qa. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6719–6728.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A discrete hard em approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2851–2864.

Devendra Singh Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. 2021. End-to-end training of neural retrievers for open-domain question answering. In *ACL/IJCNLP 2021*, pages 6648–6662. Association for Computational Linguistics.

A Appendix

NER Type	**Candidate Question Phrases**
CARDINAL	"what",
DATE	"when","what time", "what date",
EVENT	"what event","what", "which event",
FAC	"where","what buildings",
GPE	"where", "what country",
LANGUAGE	"what language","which language",
LAW	"which law","what law",
LOC	"where", "what location", "which place", "what place",
MONEY	"how much money","how much",
NORP	"what", "what groups", "where",
ORDINAL	"what rank","what",
ORG	"which organization", "what organization", "what",
PERCENT	"what percent", "what percentage",
PERSON	"who", "which person",
PRODUCT	"what", "what product",
QUANTITY	"how many", "how much",
TIME	"when", "what time",
WORK_OF_ART	"what", "what title"

Table 5: Question phrase replacement rules for different types of entities.

k-Rater Reliability:
The Correct Unit of Reliability for Aggregated Human Annotations

Ka Wong
Google Research
danicky@gmail.com

Praveen Paritosh
Google Research
pkp@google.com

Abstract

Since the inception of crowdsourcing, aggregation has been a common strategy for dealing with unreliable data. Aggregate ratings are more reliable than individual ones. However, many natural language processing (NLP) applications that rely on aggregate ratings only report the reliability of individual ratings, which is the incorrect unit of analysis. In these instances, the data reliability is under-reported, and a proposed *k-rater reliability* (kRR) should be used as the correct data reliability for aggregated datasets. It is a multi-rater generalization of *inter-rater reliability* (IRR). We conducted two replications of the WordSim-353 benchmark, and present empirical, analytical, and bootstrap-based methods for computing kRR on WordSim-353. These methods produce very similar results. We hope this discussion will nudge researchers to report kRR in addition to IRR.

1 Introduction

Crowdsourcing has become a mainstay for data collection in NLP (Geva et al., 2019; Sabou et al., 2014). It can produce data in a scalable and cost effective manner. However, these benefits come at a cost: quality. The reliability of crowd workers is always of central concern. One common strategy to increase the data reliability is to collect multiple, independent judgements and to use the aggregated judgements instead. Indeed, early papers such as Snow et al. (2008) show that average ratings correlate more strongly with expert judgements. This makes sense, as average ratings are known to have a higher reliability than individual ones (Ebel, 1951).

A number of strategies have been proposed to address data quality issues, e.g. rater modeling, label correction, label pruning (Kumar and Lease, 2011), but aggregation remains very popular (Prabhakaran et al., 2021). Sheshadri and Lease (2013) present nine crowdsourced datasets across a wide range of NLP tasks to compare different aggregation methods. See Difallah and Checco (2021) for a recent review of aggregation techniques. In short, aggregation has become the default method for acquiring reliable data from the crowd.

Interestingly, after we adopted aggregation as a community, we forgot to update our reliability measures correspondingly. The field continues to report data reliability in terms of IRR, even when aggregate ratings are used. Focusing on IRR, we are unable to capture the increase in reliability due to aggregation. The actual data reliability is hence unknown. This has important consequences. Reliability is often used as a safeguard for *reproducibility*. Therefore conclusions about the reproducibility of a dataset drawn based the reliability of individual ratings may be different than that based on the reliability of aggregate ratings.

By reporting the correct reliability that is actually higher, this may even have a side effect of lessening the stigma on low-IRR datasets. As a result, this may create a path forward towards reliable data on subjective tasks, where a high IRR is difficult to obtain, such as emotions (Wong et al., 2021) and toxicity (Wulczyn et al., 2017). With a reproducibility crisis looming in the background (Baker, 2016; Hutson, 2018), more frequent and accurate reporting of reliability is our primary safeguard (Paritosh, 2012).

We denote the reliability of aggregate ratings as *k-rater reliability* (kRR), in order to differentiate it from inter-rater reliability. In this paper we present a few methods for computing kRR. First, we demonstrate a general, empirical approach that is based on replications. To that end, we conducted two replications of WordSim-353 (Finkelstein et al., 2001), a widely used word similarity dataset. We then discuss two other alternatives that do not require replications. One is a re-sampling-based bootstrap approach (Efron and Tibshirani, 1994). It is suitable for experiments with a high rating redun-

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 378 - 384
May 22-27, 2022

dancy. The other is an existing analytical approach based on intraclass correlation (ICC). It is suitable for continuous data where the aggregation is the mean. We conclude with recommendations for reporting reliability of crowdsourced annotations, and novel research questions to expand the usefulness of kRR.

2 Related Work

Various authors have stressed the importance of measuring reliability for the correct unit of analysis. Ebel (1951) asks "Is it better to estimate the reliability of individual ratings or the reliability of average ratings? If decisions are based upon average ratings, it of course follows that the reliability with which one should be concerned is the reliability of those averages." Shrout and Fleiss (1979) and Hallgren (2012) reiterate similar points.

These studies primarily focus on the reliability of the *mean*, which is just one of many different aggregation methods. There is a reason. Not only is the mean a popular choice, it is also the only known choice where the reliability of the aggregate ratings can be computed *analytically* from the reliability of individual ratings. This is done in the ICC framework. ICC is typically used to measure the reliability of single ratings, but it actually has a variant that can be used for mean ratings as well. Shrout and Fleiss (1979) list several types of ICC coefficient, one of which is for mean ratings. They call it ICC(k), where k is the number of ratings per item. In this generalized notation, ICC(1) is just the reliability of individual ratings, or the IRR. Note that McGraw and Wong (1996) use a slightly different notation, ICC($1, k$), to explicitly denote that it is for a one-way random effects model, where the raters are treated as interchangeable. That is a common assumption in most crowdsourcing experiments done on commercial platforms such as Amazon Mechanical Turk.

ICC(k) is an established way of measuring the reliability of mean ratings, hence it is readily usable by researchers. However, it has some drawbacks. Being part of the ICC family, ICC(k) is only applicable to continuous data. In addition, ICC(k) measures the reliability of *mean* ratings, therefore it cannot accommodate other aggregation functions. In other words, for other popular data types, such as majority votes of binary data, there is no known coefficients for measuring the reliability of aggregate ratings. Other than ICC(k), the authors are not aware of any multi-rater generalization for other coefficients such as Cohen's (1960) *kappa* or Krippendorff's *alpha* (Krippendorff, 2011). We therefore take ICC(k) as an inspiration and abstract away from it to define a class of reliability that describes the reliability of aggregate ratings for any data types. We denote it kRR.

3 Contributions

- We emphasise the reliability of aggregate ratings is higher than that of individual ratings.
- We give a general definition of kRR, extending from the definition of IRR, and discuss three methods for computing it.
- We conduct two replications of the WordSim-353 benchmark to validate these methods.

4 *k*-Rater Reliability

We define kRR as the chance-adjusted agreement between replications of aggregate ratings. This definition is very similar to IRR. In fact, they only differ in terms of interpretation. kRR is identical to IRR other than that each individual rating in the IRR calculation is replaced by a k-rater aggregate rating. After all, the mathematics in IRR are agnostics to how those labels are produced.

Just like IRR, a minimum of two replications is required to calculate kRR. Given two vectors of aggregate ratings, one can calculate the reliability between them using any IRR coefficients that fit the purpose. kRR is designed to be analogous to IRR so that we can build upon the rich IRR literature and the various coefficient choices for different experimental conditions and assumptions. For example, in a binary task, if all the items are rated by two fixed but distinct groups of raters (raters from different locales), Cohen's (1960) *kappa* is a suitable choice. Whereas if the raters groups are homogeneous, and the rating scale is ordinal (e.g. Likert), then Krippendorff's *alpha* (Krippendorff, 2011) can be used. Just like IRR, kRR is a general concept and is agnostic to the choice of coefficient.

This definition of kRR can be directly operationalized by creating replications. We call this approach to calculating kRR the *empirical approach*. We demonstrate it in the next section on the WordSim-353 benchmark. The empirical approach is the most direct and most general, with the drawback that a minimum of two replications are required. We later present two narrower alternatives in Section 5 that do not require replications. The empirical results will be used as a golden reference to validate them.

4.1 Replicating the WordSim Dataset

WordSim-353 (Finkelstein et al., 2001) is a widely used benchmark for measuring a system's ability to compute similarity between two words, and has been cited over 1500 times. The dataset contains 353 word pairs. Each word pair is rated by the same 13 workers for their similarity on a scale from 1 to 10, to indicate how similar their meanings are. The 13 ratings on each word pair are then aggregated into a mean score. It is important to note that only the mean of the ratings are utilized by all the research using this dataset as a benchmark. So the unit of analysis is the aggregate of the 13 ratings, not individual ratings.

Nearly twenty years have elapsed since the creation of the WordSim dataset. It is impossible to recreate the original experimental conditions due to rater population changes. Therefore, we created two replications in order to approximate the kRR of the original dataset. Two is the minimum replication factor required for the empirical approach, though a higher replication would result in a more accurate measure of kRR.

We used the original annotation guidelines on Amazon Mechanical Turk. Raters were paid on average USD 9.5 per hour. In each replication, we collected 13 judgements on each of the same 353 word pairs. There was a detail that we did not follow. In the original experiment, the authors employed 13 unique raters, and each one rated all 353 word pairs. In our replications, we followed more modern conventions and limited the contributions of each individual rater for better generalizability. This detail aside, these are our best attempts to replicate the original experiment. The data is publicly available at `https://github.com/google-research-datasets/wordsim-replications`.

4.2 Empirical kRR Results

We take k columns of ratings at random from each of the two replications, compute the k-rater mean scores for each replication, and measure the reliability between them using Krippendorf's *alpha*, the most widely used and general reliability index. We do this for $k = 1, 2, \ldots, 13$. The resulting kRR values are shown in Fig.1. At $k = 1$, the IRR is 0.574, slightly lower than the 0.6 originally reported in Finkelstein et al. (2001). At $k = 13$, the k-rater reliability is 0.940, quite a bit higher than the IRR. In addition, Fig.1 shows the marginal returns on increasing the number of ratings on the replicated datasets.

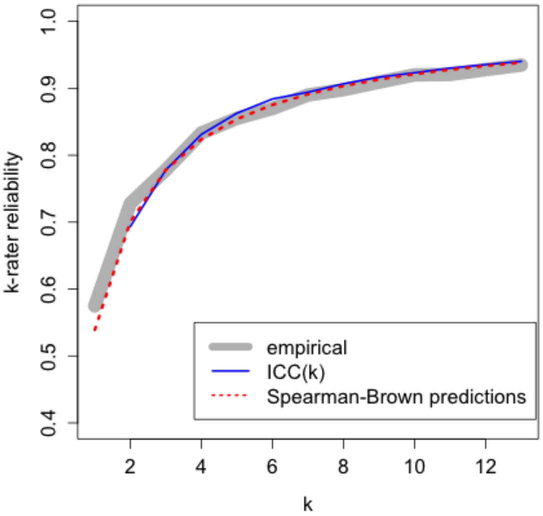


Figure 1: k-rater reliability for replications of WordSim benchmark, calculated using 3 different methods: 1) Empirical, based on replications, 2) ICC(k), analytical, and 3) SB predictions. Note ICC(1) is not available as we only have a single column of ratings available at $k = 1$. All SB predictions are based on only 2 ratings per item.

5 Other Approaches to Computing kRR

The empirical approach is general, as it can accommodate any choice of rating scale, aggregation function, and reliability coefficient. However, it has a major drawback. As we see in Section 4.1, it can be difficult to do a perfect replication post-fact. This backward incompatibility will present a challenge to computing kRR for existing datasets. Below we present two alternatives that can work on existing datasets under some conditions without requiring any additional data collection. One is a re-sampling based bootstrap approach (Efron and Tibshirani, 1994), the other is ICC(k).

5.1 Bootstrap

Bootstrap (Efron and Tibshirani, 1994) is a re-sampling technique commonly used for quantifying uncertainty in statistical parameter estimation. One can bootstrap an NLP annotations dataset by re-sampling ratings within each annotation item with replacement at the same sample size. If one treats each bootstrap sample as a replication, then one can apply the technique discussed in Section 4 to obtain a *bootstrapped* kRR. Bootstrap is an approximate technique and works better with larger sample sizes, typically 20 observations and above for a single distribution. The 13-rating redundancy in the WordSim replications is arguably small for a typical bootstrap exercise, but it makes up for it with a large number of items.

Before we apply bootstrap to the original WordSim dataset, we first verify its soundness by comparing it against the empirical results in Section 4.2. When applied to one of the two recent replications, the bootstrapped kRR is 0.943. This is comparable to the 0.940 reported in Section 4.2. We then apply bootstrap to the original WordSim dataset and find a bootstrapped kRR of 0.953 (Table 1). The exact method introduced below produces a very similar value at 0.950.

5.2 Intraclass Correlation

Intraclass correlation is a popular reliability coefficient for continuous data in behavioral and medical sciences. ICC gives researchers granular control over assumptions about the raters. For example, each annotation item can be rated by the same set of raters, or different sets of raters (interchangeability). In the former, the raters can be treated as either fixed or randomly drawn from a population. Shrout and Fleiss (1979) and McGraw and Wong (1996) give very extensive treatment on different ICC types for different rater assumptions.

In this paper, we focus on the most basic definition, one that treats raters as interchangeable. The ICC for k-rater averages is denoted as ICC(k) using McGraw and Wong's notation. The reliability of individual ratings is thus given by ICC(1). ICC(k) can be computed by summing squares of differences on the data matrix. Please see Appendix A for derivation and an illustration. Otherwise, software implementations of ICC are also widely available, e.g. in R and Python.

We first verify ICC(k)'s accuracy by comparing it against the empirical results in Section 4.2. To do that, we calculate ICC(k) for one of the two recent WordSim replications for $k = 1, 2, \ldots, 13$ and overlay the results (solid blue) over the empirical curve in Fig.1. We can see ICC(k) matches the empirical results quite well.

After verifying the technique, we compute ICC(k) on the original WordSim dataset. We report in Table 1 both ICC(1) and ICC(13) to show the increase in reliability. They are respectively 0.590 and 0.950.[1]

Unit of analysis	Method	reliability
single-rating	ICC(1)	0.590
13-rating mean	ICC(13)	0.950
13-rating mean	bootstrap	0.953

Table 1: Reliability of the original WordSim benchmark. First two rows are analytical estimates ICC(1) and ICC(13). Both computed using all 13 available ratings. Third row is a resampling-based bootstrapped estimate based on 100 bootstrap samples.

5.3 Spearman-Brown Formula

Given an experiment with a k-rating redundancy, ICC(k) quantifies the reliability of the k-rater average. If this reliability is too low, the researcher may want to increase the value of k. In this case, it would be helpful to know how additional ratings would impact reliability. This is analogous to calculating the required sample size for a given margin of error in a poll. For this purpose, the Spearman-Brown prophecy formula (Spearman, 1910; Brown, 1910) can be a useful tool. It predicts ICC(k) for any value of k based on ICC(1) in the current experiment:

$$\mathrm{ICC}(k) = \frac{k \cdot \mathrm{ICC}(1)}{1 + (k-1) \cdot \mathrm{ICC}(1)}. \qquad (1)$$

Warrens (2017) and de Vet et al. (2017) recently proved that SB and ICC(k) are indeed equivalent in expectation,[2] even though they look nothing alike and were derived in very different contexts. These findings confirm past observations that SB predicts empirical results accurately (Remmers et al., 1927). A limitation of SB is clearly that it only works with ICC. However, Fleiss and Cohen (1973) show ICC is actually equivalent to weighted-kappa with quadratic weights, so it likely has wider applicability.

To verify the formula, we apply SB to one of the two recent WordSim replications and overlay the results (dotted red) over the empirical curve obtained earlier. When computing SB, we only provide it with 2 ratings, in order to assess its predictive accuracy. That is, we first compute ICC(1) with 2 randomly drawn ratings from each word

[1] The former is computed using two-way random without interaction ICC(1), the latter two-way random without interaction ICC(13). The equivalent one-way models yield identical point estimates.

[2] The only exception is two-way mixed model with interaction (Warrens, 2017).

pair, then we plug this ICC(1) value into Eq.1 for $k = 1, 2, \ldots, 13$. The SB curve is overlaid over the empirical curve in Fig.1. We see that SB tracks the empirical results very well even at high k. This is remarkable as the empirical approach requires 26 ratings for $k = 13$, whereas SB merely requires 2 for any value of k.

6 Conclusions and Discussion

We pointed out where aggregated ratings are used, as is the case in many crowdsourced datasets, reliability of aggregate ratings is the correct accounting of data reliability. We introduced k-rater reliability (kRR) as a multi-rater extension of IRR. We emphasise the reliability of aggregate ratings is higher than that of individual ratings. We present analytical and bootstrap-based methods for computing the kRR on the original WordSim dataset. Both methods produce similar estimates for 13-rater reliability ranging from 0.940 to 0.953. We conduct two replications of the entire WordSim-353 benchmark to validate these methods. We make our replication data publicly available on GitHub.

While aggregation makes it possible to have reliable benchmarks on subjective topics, some readers may feel uneasy about increasing reliability via gathering additional ratings, as opposed to other traditional means such as improving rater guidelines. We suggest to mediate this concern by reporting both IRR and kRR. In fact, kRR is not meant to replace IRR, but rather complement it. IRR speaks to the reliability of the labeling process, whereas kRR quantifies the reliability of the aggregated data we consume. We urge researchers to report both where possible. In fact, Hallgren (2012) states, "In cases where single measures ICCs are low but average-measures ICCs are high, the researcher may report both ICCs to demonstrate this discrepancy."

This research also raises interesting questions for future research:

1. How do we derive multi-rater generalizations for coefficients other than ICC? A lot of NLP annotations are binary and multi-class. Such a generalization for majority voting would be particularly useful to the field.

2. Should we apply the Landis and Koch (1977) style of reliability cutoffs to kRR, or should kRR go by a different set of standards?

We urge researchers to report both IRR and kRR of aggregated human annotations, and for further inquiry around the above fundamental questions about reliability.

Acknowledgement

We thank Lora Aroyo and Chris Welty for sharing their WordSim replication datasets. We thank Michael Quinn and Jeremy Miles for their insightful discussions and comments. We also thank all the crowd workers for providing us with valuable annotations data.

References

Monya Baker. 2016. Reproducibility crisis. *Nature*, 533(26):353–66.

William Brown. 1910. Some experimental results in the correlation of mental abilities 1. *British Journal of Psychology, 1904-1920*, 3(3):296–322.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Henrica C.W. de Vet, Lidwine B. Mokkink, David G. Mosmuller, and Caroline B. Terwee. 2017. Spearman-brown prophecy formula and cronbach's alpha: different faces of reliability and opportunities for new applications. *Journal of Clinical Epidemiology*, 85:45–49.

Djellel Difallah and Alessandro Checco. 2021. Aggregation techniques in crowdsourcing: Multiple choice questions and beyond. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4842–4844.

Robert L Ebel. 1951. Estimation of the reliability of ratings. *Psychometrika*, 16(4):407–424.

Bradley Efron and Robert J Tibshirani. 1994. *An introduction to the bootstrap*. CRC press.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414.

Joseph L Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 33(3):613–619.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. *arXiv preprint arXiv:1908.07898*.

Kevin A Hallgren. 2012. Computing inter-rater reliability for observational data: An overview and tutorial. *Tutorials in quantitative methods for psychology*, 8(1):23–34.

Matthew Hutson. 2018. Artificial intelligence faces reproducibility crisis.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability.

Abhimanu Kumar and Matthew Lease. 2011. Modeling annotator accuracies for supervised learning. In *Proceedings of the Workshop on Crowdsourcing for Search and Data Mining (CSDM) at the Fourth ACM International Conference on Web Search and Data Mining (WSDM)*, pages 19–22.

J. Richard Landis and Gary G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–74.

David Liljequist, Britt Elfving, and Kirsti Skavberg Roaldsen. 2019. Intraclass correlation–a discussion and demonstration of basic features. *PloS one*, 14(7):e0219854.

Kenneth O McGraw and Seok P Wong. 1996. Forming inferences about some intraclass correlation coefficients. *Psychological methods*, 1(1):30.

Praveen Paritosh. 2012. Human computation must be reproducible. In *WWW 2012, Lyon.*

Vinodkumar Prabhakaran, Aida Mostafazadeh Davani, and Mark Diaz. 2021. On releasing annotator-level labels and information in datasets. In *Proceedings of The Joint 15th Linguistic Annotation Workshop (LAW) and 3rd Designing Meaning Representations (DMR) Workshop*, pages 133–138, Punta Cana, Dominican Republic. Association for Computational Linguistics.

HH Remmers, NW Shock, and EL Kelly. 1927. An empirical study of the validity of the spearman-brown formula as applied to the purdue rating scale. *Journal of Educational Psychology*, 18(3):187.

Marta Sabou, Kalina Bontcheva, Leon Derczynski, and Arno Scharl. 2014. Corpus annotation through crowdsourcing: Towards best practice guidelines. In *LREC*, pages 859–866. Citeseer.

Aashish Sheshadri and Matthew Lease. 2013. Square: A benchmark for research on computing crowd consensus. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 1.

Patrick E Shrout and Joseph L Fleiss. 1979. Intraclass correlations: uses in assessing rater reliability. *Psychological bulletin*, 86(2):420.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. 2008. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii. Association for Computational Linguistics.

Charles Spearman. 1910. Correlation calculated from faulty data. *British Journal of Psychology, 1904-1920*, 3(3):271–295.

Matthijs J Warrens. 2017. Transforming intraclass correlation coefficients with the spearman–brown formula. *Journal of clinical epidemiology*, 85:14–16.

Ka Wong, Praveen Paritosh, and Lora Aroyo. 2021. Cross-replication reliability - an empirical approach to interpreting inter-rater reliability. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7053–7065, Online. Association for Computational Linguistics.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th international conference on world wide web*, pages 1391–1399.

	Rating			
Item	1	2	$\ldots j$	$\ldots k$
1	x_{11}	x_{12}	$\ldots x_{1j}$	$\ldots x_{1k}$
2	x_{21}	x_{22}	$\ldots x_{2j}$	$\ldots x_{2k}$
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
i	x_{i1}	x_{i2}	$\ldots x_{ij}$	$\ldots x_{ik}$
.	.	.	.	.
.	.	.	.	.
n	x_{n1}	x_{n2}	$\ldots x_{nj}$	$\ldots x_{nk}$

Figure 2: A convenient data matrix and notational system for data used in calculating intra-class correlation coefficients

A Appendix on ICC(k)

ICC is a family of coefficients. It has slightly different formulations to accommodate different experimental designs. One of them, ICC(k), quantifies the reliability of average ratings based on k raters, where the raters are treated as interchangeable. We illustrate its close form calculation here. It is mainly re-expressing results from previous works on ICC calculation, such as Liljequist et al. (2019) and McGraw and Wong (1996).

ICC(k) predicates on the one-way random effects model being the data generation process. The model takes the form

$$x_{ij} = \mu + \phi_i + \epsilon_{ij},$$

where x_{ij} is the rating on item i from rater j, μ is the grand mean, ϕ_i is the mean of item i, and ϵ_{ij} is a random perturbation term. Assume a data matrix with n rows (item) and k columns (raters) with no missing data, as one shown in Fig. 2. Let

$$\bar{x}_{\cdot\cdot} = \frac{1}{nk} \sum_{j=1}^{k} \sum_{i=1}^{n} x_{ij}$$

be the sample grand mean, and

$$\bar{x}_{i\cdot} = \frac{1}{k} \sum_{j=1}^{k} x_{ij}$$

be the i^{th} sample item mean. Let

$$SSW = \sum_{j=1}^{k} \sum_{i=1}^{n} (x_{ij} - \bar{x}_{i\cdot})^2$$
$$SSB = k \sum_{i=1}^{n} (\bar{x}_{i\cdot} - \bar{x}_{\cdot\cdot})^2$$

be respectively the sum of squares due to differences *within* items and the sum of squares due to differences *between* items. Then the estimator for the variance of ϵ, σ_ϵ^2, and the estimator for the variance of ϕ, σ_ϕ^2, are respectively

$$\hat{\sigma}_\epsilon^2 = \frac{SSW}{n(k-1)}$$
$$\hat{\sigma}_\phi^2 = \frac{SSB}{k(n-1)} - \frac{\hat{\sigma}_\epsilon^2}{k}.$$

Then ICC(k) can be computed as

$$\frac{\hat{\sigma}_\phi^2}{\hat{\sigma}_\alpha^2 + \hat{\sigma}_\epsilon^2/k}.$$

If we apply the above formula to individual ratings, with $k = 1$, the resulting reliability is known as inter-rater reliability. For any $k > 1$, it is an instance of the *k-rater reliability* proposed in this paper.

An Embarrassingly Simple Method to Mitigate `und` `es` `ira` `ble` Properties of Pretrained Language Model Tokenizers

Valentin Hofmann[*‡], **Hinrich Schütze**[‡], **Janet B. Pierrehumbert**[†*]

[*]Faculty of Linguistics, University of Oxford
[†]Department of Engineering Science, University of Oxford
[‡]Center for Information and Language Processing, LMU Munich
valentin.hofmann@ling-phil.ox.ac.uk

Abstract

We introduce FLOTA (Few Longest Token Approximation), a simple yet effective method to improve the tokenization of pretrained language models (PLMs). FLOTA uses the vocabulary of a standard tokenizer but tries to preserve the morphological structure of words during tokenization. We evaluate FLOTA on morphological gold segmentations as well as a text classification task, using BERT, GPT-2, and XLNet as example PLMs. FLOTA leads to performance gains, makes inference more efficient, and enhances the robustness of PLMs with respect to whitespace noise.

1 Introduction

The first step in NLP architectures using pretrained language models (PLMs) is to map text to a sequence of tokens corresponding to input embeddings. The tokenizers used to accomplish this have been shown to exhibit various undesirable properties such as generating segmentations that blur word meaning (Bostrom and Durrett, 2020; Church, 2020; Hofmann et al., 2021) and generalizing suboptimally to new domains (Tan et al., 2020; Hong et al., 2021; Sachidananda et al., 2021).

In this paper, we propose **FLOTA** (**F**ew **Lo**ngest **T**oken **A**pproximation), a simple yet effective method to mitigate some shortcomings of PLM tokenizers. FLOTA is motivated by the following hypothesis: rather than finding a segmentation that *covers all characters* of a word but *destroys its morphological structure*, it can be more beneficial to find a segmentation that *does not cover all characters* but *preserves key aspects of the morphology*. We confirm this hypothesis in this paper.

Our study investigates three PLMs and corresponding tokenizers: BERT (base, uncased; Devlin et al., 2019), which uses WordPiece (Schuster and Nakajima, 2012; Wu et al., 2016), GPT-2 (base, cased; Radford et al., 2019), which uses byte-pair encoding (BPE; Gage, 1994; Sennrich et al., 2016), and XLNet (base, cased; Yang et al., 2019), which uses Unigram (Kudo, 2018). We find that FLOTA increases the morphological quality of all tokenizers as evaluated on human-annotated gold segmentations as well as the performance of all PLMs on a text classification challenge set.

Contributions. We introduce FLOTA, a simple yet effective method to improve the tokenization of PLMs during finetuning. FLOTA uses the vocabulary of a standard tokenizer but tries to preserve the morphological structure of words during tokenization. We show that FLOTA has three advantages compared to standard tokenization: (i) it can increase the performance of PLMs on certain tasks, sometimes substantially; (ii) it makes inference more efficient by shortening the processed token sequences; (iii) it enhances the robustness of PLMs with respect to certain types of noise in the data. All this is achieved *without requiring any additional parameters or resources* compared to vanilla PLM finetuning. We also release a text classification challenge set that can serve as a benchmark for future studies on PLM tokenizers.[1]

2 Few Longest Token Approximation

Let V be a set of tokens that constitute the vocabulary of a tokenizer. For the tokenizers discussed in this paper, V contains words, subwords, and characters. Let ϕ be a model used by the tokenizer to map text to a sequence of tokens from V.

FLOTA (Few Longest Token Approximation) discards ϕ and uses V in a modified way. Given a word w not in V, FLOTA tokenizes it by determining the longest substring $s \in V$ of w, returning s, and recursing on $w \setminus s$, the string(s) remaining when s is removed from w. We stop after k recursive calls or when the residue is null. Figure 1 provides pseudocode. For the example word *undesirable* and $k = 2$, FLOTA first searches on

[1] We make our code and data available at https://github.com/valentinhofmann/flota.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 385 - 393
May 22-27, 2022 ©2022 Association for Computational Linguistics

```
MaxSubwordSplit($w, V$)
$l = length(w)$
for $j = l$ downto 0
    for $i = 0$ to $l - j + 1$
        $s = w[i \,.\,.\, i + j]$
        if $s \in V$
            $r = w[0 \,.\,.\, i] \oplus_j w[i + j \,.\,.\, l]$
            return $s$, $r$, $i$

FlotaTokenize($w, k, V$)
$s, r, i =$ MaxSubwordSplit($w, V$)
if $k$ == 1 or $hyphen(r)$
    $F = \{\}$
    $F[i] = s$
    return $F$
$F =$ FlotaTokenize($r, k - 1, V$)
$F[i] = s$
return $F$
```

Figure 1: FLOTA pseudocode. FLOTA is based on a recursive function FLOTATOKENIZE that uses a hash table F to store the longest substring s and its index i on each recursive call. s and i are found by means of a second function MAXSUBWORDSPLIT, which also returns a residue r. In practice, to ensure correct indexing throughout different recursive calls as well as prevent using discontinuous substrings for tokenization, we compute r using an operation $\oplus_j$ that concatenates two strings by putting j (length of s) hyphens between them. The recursion stops after k recursive calls or when r only consists of hyphens (determined by a boolean function $hyphen$). The hash table returned by FLOTATOKENIZE is converted to a tokenization using a simple wrapper function that sorts the found substrings by their indices (not shown). If MAXSUBWORDSPLIT does not find a substring $s \in V$, FLOTATOKENIZE returns an empty hash table (not shown).

undesirable and finds `desirable`, then searches on `un---------` and finds `un`, then stops (since $k = 2$; it would also stop for $k > 2$ since the residue is null) and returns the tokenization `un`, `desirable`. The WordPiece tokenization, on the other hand, is `und`, `es`, `ira`, `ble`.

FLOTA is guided by the following observations: many words not in V are made up of smaller and typically more frequent elements that determine their meaning (e.g., they are derivatives such as *undesirable*); many of these elements are in V.[2] By recursively searching for the longest substrings, we hope to recover the most important meaningful elements. This is also why it makes sense to stop after k recursions: if FLOTA returns the most important meaningful elements as the first few tokens, we expect to not lose much by stopping.

Model	Tokenization	$\overline{C}$	$\overline{R}$	$\overline{M}$
BERT	FIRST	.869	.817	.664
BERT	LONGEST	.865	.797	.664
BERT	FLOTA	**.990**	**.876**	**.896**
GPT-2	FIRST	.878	.674	.625
GPT-2	LONGEST	.874	.674	.625
GPT-2	FLOTA	**.988**	**.845**	**.861**
XLNet	FIRST	.886	.820	.724
XLNet	LONGEST	902	.845	.756
XLNet	FLOTA	**.992**	**.900**	**.922**

Table 1: Morphological quality. $\overline{C}$: morphological coverage ($k = 2$); $\overline{R}$: stem recall; $\overline{M}$: full match.

3 Evaluation on Gold Segmentations

English inflection is simple, but the language has highly complex word formation, i.e., derivation and compounding (Cotterell et al., 2017; Pierrehumbert and Granell, 2018). To evaluate the morphological quality of FLOTA against the standard tokenizers, we thus focus on derivatives and compounds.

Data. Our evaluation uses CELEX (Baayen et al., 1995) and LADEC (Gagné et al., 2019), two large datasets of human-annotated gold segmentations of morphologically complex words. We merge both datasets and extract all words consisting of a prefix and a stem (prefixed derivatives), a stem and a suffix (suffixed derivatives), or two stems (compounds). We create for each PLM a subset of words where both morphological elements (i.e., stems and affixes) are in the tokenizer vocabulary, but the word itself is not in the tokenizer vocabulary. In such cases, a word needs to be segmented, and it is guaranteed that *the gold segmentation is possible* given the tokenizer vocabulary. This procedure results in 11,272, 11,253, 10,848 words for BERT, GPT-2, XLNet, respectively.

Experimental Setup. We define three metrics to analyze how closely FLOTA matches the gold segmentations. We compare against two alternative tokenization strategies: representing words as the k first tokens returned by the standard tokenizer (FIRST) and representing words as the k longest tokens returned by the standard tokenizer (LONGEST). Recall that the WordPiece tokenization of the running example *undesirable* is `und`, `es`, `ira`, `ble`. With $k = 3$, FIRST is `und`, `es`, `ira` (i.e., it simply returns the first k tokens) and LONGEST is `und`, `ira`, `ble` (i.e., it returns the k

[2] Existing tokenizers have been shown to be able to recover these elements only to a very limited extent (Bostrom and Durrett, 2020; Church, 2020; Hofmann et al., 2021).

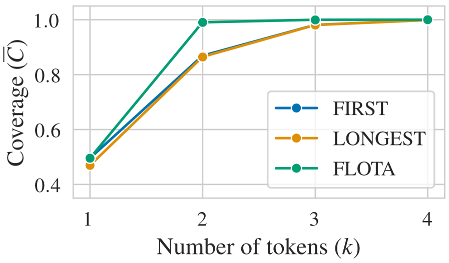


Figure 2: Morphological coverage for varying k.

longest tokens in the order in which they occur in the standard tokenization).

Morphological coverage. We analyze what proportion of morphological elements is covered by each tokenization strategy for varying k, a measure that we call *morphological coverage*, C. For *undesirable* and $k = 3$, FIRST and LONGEST contain *un* ($C = 0.5$) while FLOTA contains both *un* and *desirable* ($C = 1$). We compute the mean morphological coverage across all words, $\overline{C}$.

We find that for all three tokenizers, FLOTA already covers about 99% of the morphological elements with just $k = 2$, a value that FIRST and LONGEST only reach with $k = 4$ (Table 1, Figure 2), indicating that FLOTA needs considerably fewer tokens than the standard tokenization to convey the same amount of semantic and syntactic information. This can also be seen by examining the average number of tokens needed to fully tokenize a word (i.e., $k = \infty$), with the values for FLOTA (BERT: 2.02; GPT-2: 2.03; XLNet: 2.02) being lower than the values for the standard tokenization (BERT: 2.30; GPT-2: 2.23; XLNet: 2.26). The pairwise differences are statistically significant ($p < 0.001$) as shown by two-tailed t-tests.

Stem recall. Given its relevance for the overall lexical meaning of a word, we are interested in how often FLOTA returns the stem at $k = 1$. We test this using a measure that we call *stem recall*, R ($R = 1$ if the token is the stem,[3] otherwise $R = 0$), and compute the mean stem recall $\overline{R}$ across all words. We again compare with FIRST and LONGEST. Notice the stem according to the gold segmentation is longer than the second morphological element in 97% of the examined complex words, which means that LONGEST provides a close estimate of how often the full standard tokenization contains the stem (since any other element in the full standard tokenization is shorter and hence very unlikely to be the stem).

FLOTA returns the stem considerably more often than either FIRST or LONGEST, but there are clear differences between the models (Table 1): for GPT-2, FLOTA increases $\overline{R}$ by more than 15% while the difference amounts to 5% for XLNet.

Full match. Extending the evaluation of stem recall, we examine whether the tokenization at $k = 2$ is identical to the gold segmentation (which always has two elements) using a measure that we call *full match*, M ($M = 1$ if the tokenization exactly matches the gold segmentation, otherwise $M = 0$). We again compute the mean value $\overline{M}$ across all words. Here, the values for both FIRST and LONGEST are identical to the performance of the full standard tokenization: for the full standard tokenization to exactly match a segmentation of two elements, it must consist of two tokens, and hence it is necessarily equal to both its first two tokens and its longest two tokens.[4] Table 1 shows that FLOTA substantially improves $\overline{M}$.

The evaluation on gold segmentations indicates that FLOTA increases the morphological quality of PLM tokenizers compared to the standard tokenization and simple alternatives. We also find underlying differences in the morphological quality of the tokenizers, with BPE and Unigram lying at the negative and positive extremes, in line with prior work (Bostrom and Durrett, 2020). Our analysis shows that WordPiece lies in between.

4 Evaluation on Downstream Task

We investigate whether the enhanced quality of FLOTA tokenizations translates to performance on downstream tasks. We focus on text classification as one of the most common tasks in NLP.

Data. We create two text classification challenge sets based on ArXiv,[5] each consisting of three datasets. Specifically, for the subject areas of computer science, maths, and physics, we extract titles for the 20 most frequent subareas (e.g., *Computation and Language*). We then sample 100/1,000 titles per subarea, resulting in three text classification datasets of 2,000/20,000 titles each, which we bundle together as ArXiv-S/L. Our sampling

[3] For compounds: one of the two stems.

[4] Surprisingly, this does not hold for Unigram, which sometimes creates *separate* start-of-word tokens; e.g., the Unigram tokenization of *americanize* is _, `american`, `ize`, where _ is a start-of-word token. Notice that in such cases (with $k = 2$), LONGEST (`american`, `ize`) matches the gold segmentation while FIRST (_, `american`) does not, explaining the performance difference for XLNet.

[5] `kaggle.com/Cornell-University/arxiv`

	ArXiv-S		ArXiv-L	
Model	Dev	Test	Dev	Test
BERT	.469	.470	.674	.659
+FLOTA	**.491**	**.485**	**.675**	**.661**
GPT-2	.329	.324	.526	.507
+FLOTA	**.353**	**.382**	**.558**	**.542**
XLNet	.435	**.454**	.660	.641
+FLOTA	**.446**	.428	**.664**	**.646**

Table 2: Performance. FLOTA leads to gains in averaged F1, particularly for BERT and GPT-2. Performance breakdowns for the individual datasets forming ArXiv-S/L are provided in Appendix A.3.

ensures that ArXiv-S/L require challenging generalization from a small number of short training examples with highly complex language. See Appendix A.1 for more details.

Experimental Setup. We split the six datasets of ArXiv-S and ArXiv-L into 60% train, 20% dev, and 20% test. We then train the three PLMs with classification heads on the six train splits, once with the standard tokenizers and once with FLOTA. See Appendix A.2 for hyperparameters. For FLOTA, we treat k as an additional tunable hyperparameter. We use F1 as the evaluation metric.

Performance. The FLOTA models perform better than the models with standard tokenization, albeit to varying degrees for the three PLMs (Table 2). The difference is most pronounced for GPT-2, with FLOTA resulting in large performance gains of up to 5%. In addition, GPT-2 performs worse than the other two PLMs on all datasets, suggesting that BPE is generally not a good fit for complex language. BERT also clearly benefits from using FLOTA, particularly on ArXiv-S. Out of the three considered PLMs, XLNet obtains the smallest performance gain from using FLOTA, but it still benefits in the majority of cases.

The advantage of FLOTA mirrors the differences observed in the morphological analysis, indicating that *FLOTA helps close the morphological quality gap* between standard tokenizations and gold segmentations. Where the gap is large, gains due to FLOTA are large (GPT-2/BPE); where it is small, gains due to FLOTA are small (XLNet/Unigram). BERT/WordPiece again lies in between.

Impact of k. To test how the performance varies with k, we focus on BERT and compare the FLOTA models for $k \in \{1, 2, 3, 4\}$ with the two alternatives FIRST and LONGEST from Section 3. See Appendix A.4 for hyperparameters.

Figure 3 shows that FLOTA only drops slightly as we decrease k, with the minimum F1 at $k = 1$ (43.6%) lying less than 2% below the maximum F1 at $k = 3$ (45.4%). In contrast, FIRST and LONGEST drop substantially as we decrease k; for FIRST, the minimum F1 at $k = 1$ (38.2%) lies more than 6% below the maximum F1 at $k = 4$ (44.8%). The fact that FLOTA is more effective at preserving performance while reducing the number of tokens aligns with the observation that it covers a larger number of morphemes and hence more semantic and syntactic content than FIRST and LONGEST for small k (Section 3).

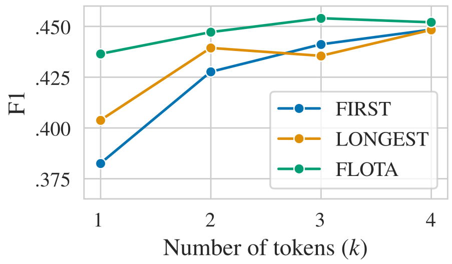


Figure 3: FLOTA is less impaired by smaller values of k (maximum number of tokens per word) than FIRST/LONGEST. Results are averaged F1 of BERT on ArXiv-S (dev/test merged).

		FLOTA			
Model	ST	$k = 1$	$k = 2$	$k = 3$	$k = 4$
BERT	12.9	8.3	10.5	11.4	11.6
GPT-2	12.9	8.3	10.7	11.5	11.8
XLNet	13.6	8.3	10.9	11.9	12.2

Table 3: Average sequence length of titles (ArXiv-L, physics). ST: standard tokenization.

Efficiency. FLOTA allows to reduce the number of tokens used to tokenize text by varying k. Since the attention mechanism scales quadratically with sequence length (Peng et al., 2021), this has beneficial effects on the computational cost involved with employing a model trained using FLOTA. We empirically find that even for $k = 4$ (the largest value used in the experiments), token sequences generated by FLOTA are on average shorter than the token sequences generated by the standard tokenizations. Table 3 shows for one dataset (ArXiv-L, physics) the average sequence length of titles encoded with the standard tokenization versus FLOTA with varying $k \in \{1, 2, 3, 4\}$ for the three PLMs.

Robustness. To examine robustness against noise, a well-known problem for PLMs (Pruthi et al., 2019), we focus on missing whitespace between words (Soni et al., 2019). We randomly drop the whitespace between two adjacent words with

	ArXiv-S (N)		ArXiv-L (N)	
Model	Dev	Test	Dev	Test
BERT	.428	.412	.579	.554
+FLOTA	**.486**	**.447**	**.652**	**.632**
GPT-2	.313	.315	.481	.463
+FLOTA	**.359**	**.357**	**.541**	**.518**
XLNet	.392	.397	.609	.589
+FLOTA	**.434**	**.421**	**.641**	**.623**

Table 4: Performance with noise (N). FLOTA clearly increases F1 on ArXiv-S/L for all PLMs when input is noisy. See Appendix A.5 for hyperparameters. Performance breakdowns for the individual datasets forming ArXiv-S/L are provided in Appendix A.6.

probability $p = 0.3$ in ArXiv-S/L. We use *unseen* noise, i.e., we only inject noise during evaluation, not training, which is the more realistic and challenging scenario (Xue et al., 2021).

The results show that synthetic noise increases the performance gap between FLOTA and standard tokenization (Table 4). While there is a drop in performance for all models compared to the experiments without noise, the drop is much more pronounced for standard tokenization; e.g., BERT's performance on ArXiv-L (test) drops by 3% with FLOTA, but by 10% without it.

5 Limitations

While we find FLOTA to work well on text classification, there are tasks for which FLOTA might prove a less suitable tokenization method: e.g., for small values of k, FLOTA often discards suffixes, which can be important for tasks with a syntactic component such as POS tagging.

Similar considerations hold for transfer to languages other than English: e.g., in the case of languages with a non-linear morphology such as Arabic, FLOTA is expected to inherit the insufficiencies of the underlying tokenizer (Alkaoud and Syed, 2020; Antoun et al., 2020).

6 Related Work

The question how PLMs are affected by their tokenizer has attracted growing interest recently. Bostrom and Durrett (2020), Church (2020), Klein and Tsarfaty (2020), and Hofmann et al. (2021) focus on the linguistic properties of tokenizers. We contribute to this line of work by conducting the first comparative analysis of all three common PLM tokenizers and releasing a challenge set as a benchmark for future studies. Another strand of research has sought to improve PLM tokenizers by training models from scratch (Clark et al., 2021; Si et al., 2021; Xue et al., 2021; Zhang et al., 2021) or modifying the tokenizer during finetuning, mostly by adding tokens and corresponding embeddings (Chau et al., 2020; Tan et al., 2020; Hong et al., 2021; Sachidananda et al., 2021). FLOTA crucially differs in that it can be used during finetuning but does not add any parameters to the PLM. Furthermore, there has been work improving tokenization by variously exploiting the probabilistic nature of tokenizers (Kudo, 2018; Provilkov et al., 2020; Cao and Rimell, 2021). By contrast, our method does not need access to the underlying model.

Our study also relates to computational work on derivational morphology (Cotterell et al., 2017; Vylomova et al., 2017; Cotterell and Schütze, 2018; Deutsch et al., 2018; Hofmann et al., 2020a,b,c) and word segmentation (Cotterell et al., 2016; Kann et al., 2016; Ruzsics and Samardžić, 2017; Mager et al., 2019, 2020; Seker and Tsarfaty, 2020; Amrhein and Sennrich, 2021). We are the first to systematically evaluate the segmentations of PLM tokenizers on human-annotated gold data.

Conceptually, the findings of our study are in line with evidence from the cognitive sciences that knowledge of a longer (i.e., more detailed and informative) sequence takes priority over any knowledge about smaller sequences (Caramazza et al., 1988; Laudanna and Burani, 1995; Baayen et al., 1997; Needle and Pierrehumbert, 2018).

7 Conclusion

We introduce FLOTA (Few Longest Token Approximation), a simple yet effective method to improve the tokenization of pretrained language models (PLMs). FLOTA uses the vocabulary of a standard tokenizer but tries to preserve the morphological structure of words during tokenization. FLOTA leads to performance gains, makes inference more efficient, and substantially enhances the robustness of PLMs with respect to whitespace noise.

Acknowledgements

This work was funded by the European Research Council (#740516) and the Engineering and Physical Sciences Research Council (EP/T023333/1). The first author was also supported by the German Academic Scholarship Foundation and the Arts and Humanities Research Council. We thank the reviewers for their helpful comments.

Ethical Considerations

FLOTA shortens the average length of sequences processed by PLMs, thus reducing their energy requirements, a desirable property given their otherwise detrimental environmental footprint (Schwartz et al., 2019; Strubell et al., 2019).

References

Mohamed Alkaoud and Mairaj Syed. 2020. On the importance of tokenization in Arabic embedding models. In *Arabic Natural Language Processing Workshop (WANLP) 5*.

Chantal Amrhein and Rico Sennrich. 2021. How suitable are subword segmentation strategies for translating non-concatenative morphology? In *Findings of the Association for Computational Linguistics: EMNLP 2021*.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT)*.

R. Harald Baayen, Ton Dijkstra, and Robert Schreuder. 1997. Singulars and plurals in Dutch: Evidence for a parallel dual-route model. *Journal of Memory and Language*, 37:94–117.

R. Harald Baayen, Richard Piepenbrock, and Leon Gulikers. 1995. *The CELEX lexical database (CD-ROM)*. Linguistic Data Consortium, Philadelphia, PA.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Kris Cao and Laura Rimell. 2021. You should evaluate your language model on marginal likelihood over tokenisations. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2021*.

Alfonso Caramazza, Alessandro Laudanna, and Cristina Romani. 1988. Lexical access and inflectional morphology. *Cognition*, 28(297-332).

Ethan C. Chau, Lucy H. Lin, and Noah A. Smith. 2020. Parsing with multilingual BERT, a small corpus, and a small treebank. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Kenneth Church. 2020. Emerging trends: Subwords, seriously? *Natural Language Engineering*, 26(3):375–382.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. 2021. CANINE: Pre-training an efficient tokenization-free encoder for language representation. In *arXiv 2103.06874*.

Ryan Cotterell and Hinrich Schütze. 2018. Joint semantic synthesis and morphological analysis of the derived word. *Transactions of the Association for Computational Linguistics*, 6:33–48.

Ryan Cotterell, Tim Vieira, and Hinrich Schütze. 2016. A joint model of orthography and morphological segmentation. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT) 2016*.

Ryan Cotterell, Ekaterina Vylomova, Huda Khayrallah, Christo Kirov, and David Yarowsky. 2017. Paradigm completion for derivational morphology. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2017*.

David Crystal. 1997. *The Cambridge encyclopedia of the English language*. Cambridge University Press, Cambridge, UK.

Daniel Deutsch, John Hewitt, and Dan Roth. 2018. A distributional and orthographic aggregation model for English derivational morphology. In *Annual Meeting of the Association for Computational Linguistics (ACL) 56*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HTL) 2019*.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38.

Christina L. Gagné, Thomas L. Spalding, and Daniel Schmidtke. 2019. LADEC: The large database of English compounds. *Behavior Research Methods*, 51(5):2152–2179.

Valentin Hofmann, Janet B. Pierrehumbert, and Hinrich Schütze. 2020a. DagoBERT: Generating derivational morphology with a pretrained language model. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2020*.

Valentin Hofmann, Janet B. Pierrehumbert, and Hinrich Schütze. 2020b. Predicting the growth of morphological families from social and linguistic factors. In *Annual Meeting of the Association for Computational Linguistics (ACL) 58*.

Valentin Hofmann, Janet B. Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Improving BERT's interpretations of complex words with derivational morphology. In *Annual Meeting of the Association for Computational Linguistics (ACL) 59*.

Valentin Hofmann, Hinrich Schütze, and Janet B. Pierrehumbert. 2020c. A graph auto-encoder model of derivational morphology. In *Annual Meeting of* *the Association for Computational Linguistics (ACL) 58*.

Jimin Hong, Taehee Kim, Hyesu Lim, and Jaegul Choo. 2021. AVocaDo: Strategy for adapting vocabulary to downstream domain. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2021*.

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2016. Neural morphological analysis: Encoding-decoding canonical segments. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2016*.

Diederik P. Kingma and Jimmy L. Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR) 3*.

Stav Klein and Reut Tsarfaty. 2020. Getting the ##life out of living: How adequate are word-pieces for modelling complex morphology? In *Workshop on Computational Research in Phonetics, Phonology, and Morphology (SIGMORPHON) 17*.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Annual Meeting of the Association for Computational Linguistics (ACL) 56*.

Alessandro Laudanna and Cristina Burani. 1995. Distributional properties of derivational affixes: Implications for processing. In Laurie B. Feldman, editor, *Morphological aspects of language processing*, pages 345–364. Lawrence Erlbaum, Hillsdale, NJ.

Manuel Mager, Özlem Çetinoğlu, and Katharina Kann. 2019. Subword-level language identification for intra-word code-switching. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HTL) 2019*.

Manuel Mager, Özlem Çetinoğlu, and Katharina Kann. 2020. Tackling the low-resource challenge for canonical segmentation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2020*.

Jeremy M. Needle and Janet B. Pierrehumbert. 2018. Gendered associations of english morphology. *Journal of the Association for Laboratory Phonology*, 9(1):119.

Hao Peng, Jungo Kasai, Nikolaos Pappas, Dani Yogatama, Zhaofeng Wu, Lingpeng Kong, Roy Schwartz, and Noah A. Smith. 2021. ABC: Attention with bounded-memory control. In *arXiv 2110.02488*.

Janet B. Pierrehumbert and Ramon Granell. 2018. On hapax legomena and morphological productivity. In *Workshop on Computational Research in Phonetics, Phonology, and Morphology (SIGMORPHON) 15*.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Annual Meeting of the Association for Computational Linguistics (ACL) 58*.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Annual Meeting of the Association for Computational Linguistics (ACL) 57*.

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

Tatyana Ruzsics and Tanja Samardžić. 2017. Neural sequence-to-sequence learning of internal word structure. In *Conference on Computational Natural Language Learning (CoNLL) 21*.

Vin Sachidananda, Jason S. Kessler, and Yi-An Lai. 2021. Efficient domain adaptation of language models via adaptive tokenization. In *Workshop on Simple and Efficient Natural Language Processing (SustaiNLP) 2*.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 37*.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2019. Green AI. In *arXiv 1907.10597*.

Amit Seker and Reut Tsarfaty. 2020. A pointer network architecture for joint morphological segmentation and tagging. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Annual Meeting of the Association for Computational Linguistics (ACL) 54*.

Chenglei Si, Zhengyan Zhang, Yingfa Chen, Fanchao Qi, Xiaozhi Wang, Zhiyuan Liu, and Maosong Sun. 2021. SHUOWEN-JIEZI: Linguistically informed tokenizers for Chinese language model pretraining. In *arXiv 2106.00400*.

Sandeep Soni, Lauren F. Klein, and Jacob Eisenstein. 2019. Correcting whitespace errors in digitized historical texts. In *Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature 3*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Annual Meeting of the Association for Computational Linguistics (ACL) 57*.

Samson Tan, Shafiq Joty, Lav R. Varshney, and Min-Yen Kan. 2020. Mind your inflections! Improving NLP for non-standard Englishes with base-inflection encoding. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2020*.

Ekaterina Vylomova, Ryan Cotterell, Timothy Baldwin, and Trevor Cohn. 2017. Context-aware prediction of derivational word-forms. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL) 15*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc Le V, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. In *arXiv 1609.08144*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2021. ByT5: Towards a token-free future with pre-trained byte-to-byte models. In *arXiv 2105.13626*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems (NeurIPS) 33*.

Xinsong Zhang, Pengshuai Li, and Hang Li. 2021. AMBERT: A pre-trained language model with multi-grained tokenization. In *Findings of the Association for Computational Linguistics: ACL 2021*.

A Appendix

A.1 Preprocessing

We exclude texts written in a language other than English and lowercase all words. We exclude titles with less than three and more than ten words. For each title, we compute the proportion of words starting with a productive prefix from the list provided by Crystal (1997). During sampling, we then weight titles by this proportion in order to make the language contained within the datasets as complex and challenging as possible.

A.2 Hyperparameters

The vocabulary size is 28,996 for BERT, 50,257 for GPT-2, and 32,000 for XLNet. The number of trainable parameters is 109,497,620 for BERT, 124,455,168 for GPT-2, and 117,324,308 for XLNet. The classification head for all three models uses softmax as the activation function.

We use a batch size of 64 and perform grid search for the number of epochs $n \in \{1, \dots, 20\}$ and the learning rate $l \in \{1 \times 10^{-5}, 3 \times 10^{-5}, 1 \times 10^{-4}\}$ (selection criterion: F1). We tune l on ArXiv-L (physics) and use the best configuration on all datasets. For the FLOTA models, we additionally tune $k \in \{1, 2, 3, 4\}$ (selection criterion: F1). Models are trained with categorical cross-entropy as the loss function and Adam (Kingma and Ba, 2015) as the optimizer. Experiments are performed on a GeForce GTX 1080 Ti GPU (11GB).

A.3 Performance

Table 5 provides breakdowns of the performance for the individual datasets forming ArXiv-S/L.

A.4 Hyperparameters

All hyperparameters are as for the main experiment (see Appendix A.2). For the learning rate, we use the best configuration from the main experiment. For FIRST and LONGEST, we tune $k \in \{1, 2, 3, 4\}$ (selection criterion: F1), identically to FLOTA in the main experiment.

A.5 Hyperparameters

All hyperparameters are as for the main experiment (see Appendix A.2). For the learning rate, we use the best configuration from the main experiment.

A.6 Performance

Table 6 provides breakdowns of the performance for the individual datasets forming ArXiv-S/L.

	ArXiv-S						ArXiv-L					
	Dev			Test			Dev			Test		
Model	CS	MATH	PHYS	CS	MATH	PHYS	CS	MATH	PHYS	CS	MATH	PHYS
BERT	.546	.358	.502	.498	.407	.504	.682	.660	.679	.649	.653	.675
+FLOTA	.546	.414	.514	.483	.404	.567	.677	.663	.686	.652	.658	.672
GPT-2	.354	.281	.353	.316	.261	.395	.493	.506	.578	.465	.498	.559
+FLOTA	.348	.313	.398	.370	.323	.454	.520	.549	.603	.498	.540	.587
XLNet	.473	.357	.476	.489	.358	.515	.654	.643	.684	.627	.642	.655
+FLOTA	.450	.402	.486	.415	.346	.522	.660	.651	.681	.633	.641	.665

Table 5: Performance (F1). CS: computer science; MATH: mathematics; PHYS: physics.

	ArXiv-S (N)						ArXiv-L (N)					
	Dev			Test			Dev			Test		
Model	CS	MATH	PHYS	CS	MATH	PHYS	CS	MATH	PHYS	CS	MATH	PHYS
BERT	.479	.333	.470	.566	.566	.605	.417	.338	.481	.531	.544	.588
+FLOTA	.548	.400	.511	.652	.640	.664	.452	.372	.518	.631	.620	.644
GPT-2	.336	.261	.342	.452	.461	.530	.326	.252	.366	.423	.454	.511
+FLOTA	.358	.316	.402	.514	.527	.582	.370	.296	.405	.481	.511	.562
XLNet	.431	.311	.433	.607	.594	.625	.470	.300	.421	.587	.576	.605
+FLOTA	.432	.398	.474	.646	.623	.655	.435	.360	.466	.627	.612	.631

Table 6: Performance (F1) with noise (N). CS: computer science; MATH: mathematics; PHYS: physics.

SCD: Self-Contrastive Decorrelation for Sentence Embeddings

Tassilo Klein
SAP AI Research
tassilo.klein@sap.com

Moin Nabi
SAP AI Research
m.nabi@sap.com

Abstract

In this paper, we propose Self-Contrastive Decorrelation (SCD), a self-supervised approach. Given an input sentence, it optimizes a joint self-contrastive and decorrelation objective. Learning a representation is facilitated by leveraging the contrast arising from the instantiation of standard dropout at different rates. The proposed method is conceptually simple yet empirically powerful. It achieves comparable results with state-of-the-art methods on multiple benchmarks without using contrastive pairs. This study opens up avenues for efficient self-supervised learning methods that are more robust than current contrastive methods.[1]

1 Introduction

Unsupervised learning of representation (a.k.a. embedding) is a fundamental problem in NLP and has been studied extensively in the literature (Mikolov et al., 2013; Pennington et al., 2014; McCann et al., 2017; Peters et al., 2018). Sentence embeddings are essential for numerous language processing applications, such as machine translation, sentiment analysis, information retrieval, and semantic search. Recently, self-supervised pre-training schemes have been successfully used in the context of transformer architectures, leading to a paradigm shift in natural language processing and understanding (Devlin et al., 2018; Liu et al., 2019; Radford et al., 2018) The idea here is to employ an auxiliary task, which enforces an additional objective during training. Typically, this entails predictions based on a subset of information from the context. Most objectives found effective in practice are quite simple. Some successful examples of such pretext tasks are Masked Language Model (MLM), Next Sentence Prediction (NSP), Sentence Order Prediction (SOP), etc. (Devlin et al., 2019; Liu et al., 2019; Lan et al., 2019). When working with unlabeled data, contrastive learning is among the most powerful approaches in self-supervised learning. The goal of contrastive representation learning is to learn an embedding space in such a manner that similar sample pairs (i.e., *positive pairs*) stay close to each other. Simultaneously, dissimilar sample pairs (i.e., *negative pairs*) are far pushed apart. To this end, different augmented views of the same sample and the augmented views from different samples are used as positive and negative pairs. These methods have shown impressive results over a wide variety of tasks from visual to textual representation learning (Chen et al., 2020a,b; Gao et al., 2021; Grill et al., 2020; Chen and He, 2021).

Different techniques have been proposed for the augmentation and selection of positive and negative pairs. For example, DeCLUTR (Giorgi et al., 2021) proposes to take different spans from the same document as positive pairs, while CT (Carlsson et al., 2020) aligns embeddings of the same sentence from two different encoders. CERT (Fang et al., 2020) applies the back-translation to create augmentations of original sentences, and IS-BERT (Zhang et al., 2020) maximizes the agreement between global and local features. Finally, CLEAR (Wu et al., 2020) employs multiple sentence-level augmentation strategies to learn a sentence representation. Despite the simplicity of these methods, they require careful treatment of negative pairs, relying on large batch sizes (Chen et al., 2020a) or sophisticated memory strategies. These include memory banks (Chen et al., 2020b; He et al., 2020) or customized mining strategies (Klein and Nabi, 2020) to retrieve negative pairs efficiently. In NLP specifically, the endeavor of "hard negative mining" becomes particularly challenging in the unsupervised scenario. Increasing training batch size or the memory bank size implicitly introduces more hard negative samples, coming along with the heavy burden of large memory requirements.

[1] Source code and pre-trained models are available at: https://github.com/SAP-samples/acl2022-self-contrastive-decorrelation/

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 394 - 400
May 22-27, 2022 ©2022 Association for Computational Linguistics

In this paper, we introduce SCD, a novel algorithm for self-supervised learning of sentence embedding. SCD achieves comparable performance in terms of sentence similarity-based tasks compared with state-of-the-art contrastive methods *without*, e.g., employing explicit contrastive pairs. Rather, in order to learn sentence representations, the proposed approach leverages the *self*-contrast imposed on the augmentations of a *single* sample. In this regard, the approach builds upon the idea that sufficiently strong perturbation of the sentence embedding reflects the semantic variations of the sentence. However, it is unclear which perturbation is simply a slight variation of the sentence without changing the semantic (positive pair) and which perturbation sufficiently modifies the semantic to create a negative sample. Such ambiguity manifests itself in the augmented sample sharing the characteristics of both negative and positive samples. To accommodate this, we propose an objective function consisting of two opposing terms, which acts on augmentations pairs of a sample: **i)** self-contrastive divergence (*repulsion*), and **ii)** feature decorrelation (*attraction*). The first term treats the two augmentations as a negative pair pushing apart the different views. In contrast to that, the second term attends to the augmentations as a positive pair. Thus, it maximizes the correlation of the same feature across the views, learning invariance w.r.t. the augmentation. Given the opposing nature of the objectives, integrating them in a joint loss yields a min-max optimization scheme. The proposed approach avoids degenerated embeddings by framing the representation learning objective as an attraction-repulsion trade-off. Simultaneously, it learns to improve the semantic expressiveness of the representation. Due to the difficulty of augmentation in NLP, the proposed approach generates augmentation "on-the-fly" for each sample in the batch. To this end, multiple augmentations are produced by *varying* dropout rates for each sample. We empirically observed that SCD is more robust to the choice of augmentations than pairwise contrastive methods; we believe that not relying on contrastive pairs is one of the main reasons for this, an observation also made in self-supervised learning literature such as BYOL (Grill et al., 2020). While other methods take different augmentation or different copies of models, we utilized the different outputs of the same sentence from standard dropout.

Most related to our paper is (Gao et al., 2021), which considers using dropout as data augmentation in the context of contrastive learning. A key novelty of our approach is that we use the dropout for creating the *self*-contrastive pairs, which can be utilized as *both* positive and negative. At last, we note that our model is different from the *pairwise* feature decorrelation or whitening in (Zbontar et al., 2021; Su et al., 2021; Ermolov et al., 2021), which encourage similar representations between augmented views of a sample while minimizing the redundancy within the representation vector. A key difference compared to these methods is that they ignore the contrastive objective completely. In contrast, our method takes it into account and provides the means to treat self-contrastive views as positive and negative pairs simultaneously.

Our contribution: i) generation of sentence embeddings by leverage multi-dropout **ii)** elimination of reliance on negative pairs using self-contrast, **iii)** proposing feature decorrelation objective for non-contrastive self-supervised learning in NLP.

2 Method

Our approach relies on the generation of two views A and B of samples. To this end, augmentations are generated in embedding space for each sample x_i in batch X. Batches are created from samples of set $\mathcal{D} = \{(x_i)\}_{i=1}^{N}$, where N denotes the number of sample (sentences). Augmentations are produced by an encoder f_θ, parametrized by θ. The output of the encoder is the embeddings of samples in X denoted as $H^A \in \mathcal{T}$ and $H^B \in \mathcal{T}$. Here $\mathcal{T}$ denotes the embedding space. Next, we let, $\boldsymbol{h}_i \in \mathcal{T}$ denote the associated representation of the sentence. The augmentation embeddings produced per sample are then denoted $\boldsymbol{h}_i^A$ and $\boldsymbol{h}_i^B$. To obtain the different embedding, we leverage a transformer language model as an encoder in combination with *varying* dropout rates. Specifically, one augmentation is generated with *high* dropout and one with *low* dropout. This entails employing different random masks during the encoding phase. The random masks are associated with *different* ratios, r_A and r_B, with $r_A < r_B$. Integrating the distinct dropout rates into the encoder, we yield $\boldsymbol{h}_i^A = f_\theta(x_i, r_A)$ and $\boldsymbol{h}_i^B = f_\theta(x_i, r_B)$. Given the embeddings, we leverage a joint loss, consisting of two objectives:

$$\min_{\theta_1, \theta_2} \mathcal{L}_S(f_{\theta_1}) + \alpha \mathcal{L}_C(f_{\theta_1}, p_{\theta_2}) \tag{1}$$

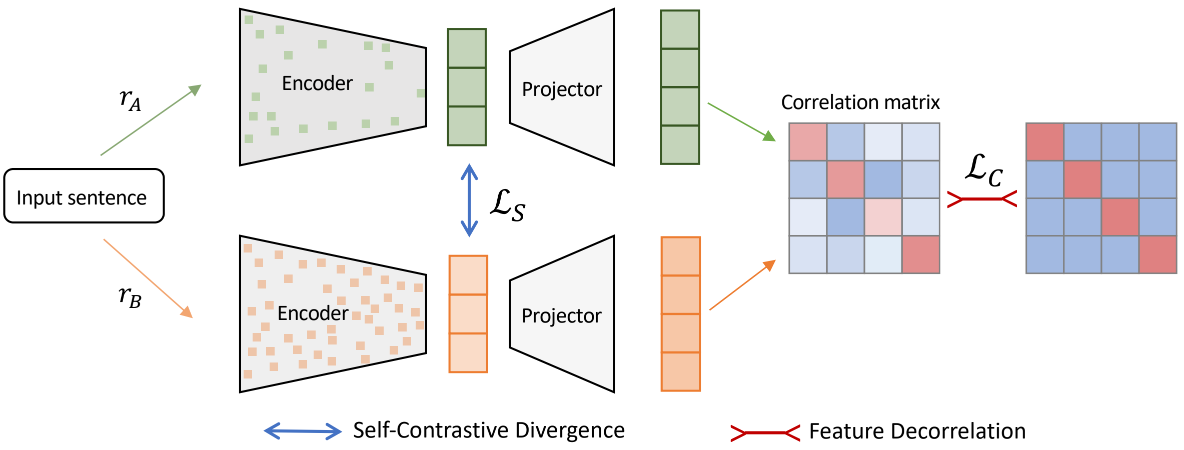


FIGURE 1. Schematic illustration of the proposed approach (best shown in color). Starting from an input sentence **(left)**, two embeddings are produced by varying the dropout-rate in the encoder. Patches within the encoder indicate masking due to dropout. Different dropout rates and resulting embeddings color-coded: **low dropout**, **high dropout**. Self-contrastive loss is imposed on the embeddings **(center)**. A projector maps embeddings to to a high-dimensional feature space, where the features are decorrelated **(right)**.

Here $\alpha \in \mathbb{R}$ denotes a hyperparameter and $p : \mathcal{T} \rightarrow \mathcal{P}$ is a projector (MLP) parameterized by θ_2, which maps the embedding to $\mathcal{P}$, with $|\mathcal{P}| \gg |\mathcal{T}|$. The objective of $\mathcal{L}_S$ is to increase the contrast of the augmented embedding, pushing apart the embeddings $\boldsymbol{h}_i^A$ and $\boldsymbol{h}_i^B$. The objective of $\mathcal{L}_C$ is to reduce the redundancy and promote invariance w.r.t. augmentation in a high-dimensional space $\mathcal{P}$. See Fig. 1 for a schematic illustration of the method.

2.1 Self-Contrastive Divergence:

Self-contrast seeks to create a contrast between the embeddings arising from different dropouts. Hence, $\mathcal{L}_S$ consists of the cosine similarity of the samples in the batch as:

$$\mathcal{L}_S = \frac{1}{N}\sum_i^N \boldsymbol{h}_i^A \cdot (\boldsymbol{h}_i^B)^T \left(\|\boldsymbol{h}_i^A\|\|\boldsymbol{h}_i^B\|\right)^{-1} \quad (2)$$

2.2 Feature Decorrelation:

$\mathcal{L}_C$ seeks to make the embeddings invariant to augmentation while at the same time reducing the redundancy in feature representation. To this end, the embedding $\boldsymbol{h}_i$ is projected up from $\mathcal{T}$ to a high-dimensional space $\mathcal{P}$, where decorrelation is performed. To avoid clutter in notation, we let $p_i^* = p(h_i^*)$ and $* \in \{A, B\}$, denote the augmented embedding vectors of sample x_i after applying a projection with $p(.)$. Then, a correlation matrix is computed from the projected embeddings. Its entries $C_{j,k}$ are:

$$C_{jk} = \sum_i p_{i,j}^A \cdot p_{i,k}^B \left(\sum_i (p_{i,j}^A)^2 (p_{i,k}^B)^2\right)^{-\frac{1}{2}} \quad (3)$$

Here, $p_{i,j}^* \in \mathbb{R}$ denotes the j^{th} component in the projected embedding vector. Then the loss objective for feature decorrelation is defined as:

$$\mathcal{L}_C = -\sum_j (1 - C_{jj})^2 + \lambda \sum_j \sum_{j \neq k} C_{jk}^2 \quad (4)$$

The first term seeks to achieve augmentation invariance by maximization of the cross-correlation along the diagonal. The second term seeks to reduce redundancy in feature representation by minimizing correlation beyond the diagonal. Given that these objectives are opposing, $\lambda \in \mathbb{R}$ is a hyperparameter, controlling the trade-off.

3 Experiments & Results

3.1 Training Setup:

Training is started from a pre-trained transformer LM. Specifically, we employ the Hugging Face (Wolf et al., 2020) implementation of BERT and RoBERTa. For sentence representation, we take the embedding of the `[CLS]` token. Then similar to (Gao et al., 2021), we train the model in an unsupervised fashion on 10^6 randomly samples sentences from Wikipedia. The LM is trained with a learning rate of $3.0\mathrm{e}{-5}$ for 1 epoch at batch-size of 192. The projector MLP q has three linear layers, each with 4096 output units in conjunction with ReLU and BatchNorm in between. For BERT hyperparameters are $\alpha = 0.005$, $\lambda = 0.013$, and dropout rates are $r_A = 5.0\%$ and $r_B = 15.0\%$. For RoBERTa hyperparameters are $\alpha = 0.0033$, $\lambda = 0.028$, and dropout rates are $r_A = 6.5\%$ and

Semantic Textual Similarity (STS) Benchmark								
Model	**STS12**	**STS13**	**STS14**	**STS15**	**STS16**	**STS-B**	**SICK-R**	**Avg.**
GloVe embeddings (avg.)$^{\clubsuit}$	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
BERT$_{\texttt{base}}$ `[CLS]`-embedding	21.54	32.11	21.28	37.89	44.24	20.29	42.42	31.40
BERT$_{\texttt{base}}$(first-last avg)$^{\diamondsuit}$	39.70	59.38	49.67	66.03	66.19	53.87	62.06	56.70
BERT$_{\texttt{base}}$-flow$^{\diamondsuit}$	58.40	67.10	60.85	75.16	71.22	68.66	64.47	66.55
BERT$_{\texttt{base}}$-whitening$^{\diamondsuit}$	57.83	66.90	60.90	75.08	71.31	68.24	63.73	66.28
IS-BERT$_{\texttt{base}}$$^{\heartsuit}$	56.77	69.24	61.21	75.23	70.16	69.21	64.25	66.58
CT-BERT$_{\texttt{base}}$$^{\diamondsuit}$	61.63	76.80	68.47	77.50	76.48	74.31	69.19	72.05
SimCSE-BERT$_{\texttt{base}}$	**68.05**	**80.38**	**72.62**	**78.96**	**76.90**	75.11	69.37	**74.48**
$*$ SCD-BERT$_{\texttt{base}}$	66.94	78.03	69.89	78.73	76.23	**76.30**	**73.18**	74.19
RoBERTa$_{\texttt{base}}$ `[CLS]`-embedding	16.67	45.56	30.36	55.08	56.99	38.82	61.89	43.62
RoBERTa$_{\texttt{base}}$(first-last avg)$^{\diamondsuit}$	40.88	58.74	49.07	65.63	61.48	58.55	61.63	56.57
SimCSE-RoBERTa$_{\texttt{base}}$	**67.05**	**80.01**	**70.93**	79.66	**80.06**	**78.38**	68.30	**74.91**
$*$ SCD-RoBERTa$_{\texttt{base}}$	63.53	77.79	69.79	**80.21**	77.29	76.55	**72.10**	73.89

TABLE 1. Sentence embedding performance on STS tasks measured as Spearman's correlation. ♣: results from Reimers and Gurevych (2019); ♡: results from Zhang et al. (2020); ◇ results from Gao et al. (2021); other results are by ourselves. Dashed line (- -), separates BERT (upper part) and RoBERTa (lower part) language models.

$r_B = 24.0\%$. The values were obtained by grid-search. First a coarse-grid was put in place with a step-size of 0.1 for α, 10% for the dropout rates r_A, r_B. For λ the coarse-grid consisted of different magnitudes $\{0.1, 0.01, 0.001\}$. Second, on a fine-grid with step-size of 0.01 and 1%, respectively.

3.2 Evaluation Setup:

Experiments are conducted on 7 standard semantic textual similarity (STS) tasks. In addition to that, we also evaluate on 7 transfer tasks. Specifically, we employ the SentEval toolkit (Conneau and Kiela, 2018) for evaluation. As proposed by (Reimers and Gurevych, 2019; Gao et al., 2021), we take STS results as the main comparison of sentence embedding methods and transfer task results for reference. For the sake of comparability, we follow the evaluation protocol of (Gao et al., 2021), employing Spearman's rank correlation and aggregation on all the topic subsets.

3.3 Main Results

3.3.1 Semantic Textual Similarity:

We evaluate on 7 STS tasks: (Agirre et al., 2012, 2013, 2014, 2015, 2016), STS Benchmark (Cer et al., 2017) and SICK-Relatedness (Marelli et al., 2014). These datasets come in sentence pairs together with correlation labels in the range of 0 and 5, indicating the semantic relatedness of the pairs. Results for the sentence similarity experiment can be seen in Tab. 1. The proposed approach is on-par with state-of-the-art approaches. Using BERT-LM, we outperform the next-best approach on STS-B (**+1.19**) and on SICK-R (**+3.81**) points. Using RoBERTa-LM, we outperform the next best comparable approach (SimCSE-RoBERTA$_{\texttt{base}}$) on STS-15 (**+0.55%**) and SICK-R (**+3.8%**).

3.3.2 Transfer task:

We evaluate our models on the following transfer tasks: MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), SUBJ (Pang and Lee, 2004), MPQA (Wiebe et al., 2005), SST-2 (Socher et al., 2013), TREC (Voorhees and Tice, 2000) and MRPC (Dolan and Brockett, 2005). To this end, a logistic regression classifier is trained on top of (frozen) sentence embeddings produced by different methods. We follow default configurations from SentEval. Results for the transfer task experiment can be seen in Tab. 2. SCD is on-par

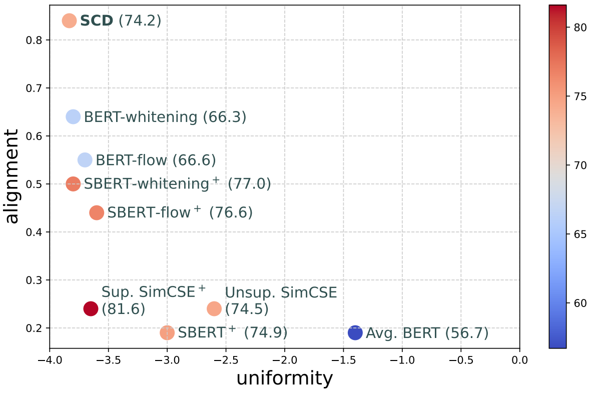


FIGURE 2. Quantitative analysis of embeddings - *alignment* vs. *uniformity* (the smaller, the better). Points represent average STS performance using BERT$_{\texttt{base}}$, with Spearman's correlation color coded (+ corresponds to supervised methods).

Transfer Benchmark								
Model	**MR**	**CR**	**SUBJ**	**MPQA**	**SST**	**TREC**	**MRPC**	**Avg.**
GloVe embeddings (avg.)♣	77.25	78.30	91.17	87.85	80.18	83.00	72.87	81.52
Skip-thought♡	76.50	80.10	93.60	87.10	82.00	92.20	73.00	83.50
Avg. BERT embeddings♣	78.66	86.25	94.37	88.66	84.40	**92.80**	69.54	84.94
BERT `[CLS]`-embedding	**81.83**	**87.39**	95.48	88.21	**86.49**	91.00	72.29	**86.10**
IS-BERT$_{\texttt{base}}$♡	81.09	87.18	94.96	**88.75**	85.96	88.64	74.24	85.83
SimCSE-BERT$_{\texttt{base}}$	80.74	85.75	93.96	88.60	84.57	86.20	73.51	84.76
$*$ SCD-BERT$_{\texttt{base}}$	73.21	85.80	**99.56**	88.67	85.89	89.80	**75.71**	85.52
RoBERTa `[CLS]`-embedding	81.27	84.77	94.15	84.18	86.71	81.20	72.17	83.49
SimCSE-RoBERTa$_{\texttt{base}}$	65.00	87.28	**99.60**	**86.63**	87.26	80.80	72.23	82.69
$*$ SCD-RoBERTa$_{\texttt{base}}$	**82.17**	**87.76**	93.67	85.69	**88.19**	**83.40**	**76.23**	**85.30**

TABLE 2. Transfer task result measured as accuracy. ♣: results from Reimers and Gurevych (2019); ♡: results from Zhang et al. (2020); ♢ results from Gao et al. (2021); other results are by ourselves. Dashed line (- -), separates BERT (upper part) and RoBERTa (lower part) language models.

with state-of-the-art approaches. Using BERT-LM, we outperform the next best approach on SUBJ (**+4.6%**) and MRPC (**+2.2%**). Using RoBERTa-LM, we outperform the next best comparable approach (SimCSE-RoBERTA$_{\texttt{base}}$) on almost all benchmarks, with an average margin of (**+2.61%**).

3.4 Analysis

3.4.1 Ablation Study:

We evaluated each component's performance by removing them individually from the loss to assess both loss terms' contributions. It should be noted that $\mathcal{L}_S$ of Eq. 2 and $\mathcal{L}_C$ of Eq. 4 both interact in a competitive fashion. Hence, only the equilibrium of these terms yields an optimal solution. Changes - such as eliminating a term - have detrimental effects, as they prevent achieving such an equilibrium, resulting in a significant drop in performance. See Tab. 3 for the ablation study on multiple benchmarks. Best performance is achieved in the presence of all loss terms.

3.4.2 Uniformity and Alignment Analysis:

To better understand the strong performance of SCD, we borrow the analysis tool from (Wang and Isola, 2020), which takes *alignment* between semantically-related positive pairs and *uniformity* of the whole representation space to measure the quality of learned embeddings. Figure 2 shows *uniformity* and *alignment* of different methods and their results on the STS. SCD achieves the best in terms of *uniformity*, reaching to the supervised counterparts (**-3.83**), which can be related to the strong effect of the self-contrastive divergence objective. It shows the *self*-contrastive pairs can effectively compensate for the absence of contrastive pairs. In terms of *alignment*, SCD is inferior to other counterparts (**0.84**), which can be attributed to the fact that our repulsion objective mainly focuses on the feature decorrelation aiming to learn a more effective and efficient representation. This is reflected in the final results on the STS where SCD obtains significantly higher correlation even compared to the method with lower *alignment* such as BERT-whitening or BERT-flow.

Method	**STS**	**STS-B**	**SICK-R**
BERT-$_{\texttt{base}}$	31.41	20.29	42.42
SCD ($\mathcal{L}_S$)	35.70	23.59	49.88
SCD ($\mathcal{L}_C$)	66.48	67.57	67.97
SCD ($\mathcal{L}_S$+$\mathcal{L}_C$)	**73.96**	**76.30**	**73.18**

TABLE 3. Ablation study, performance in average Spearman correlation on Semantic Texual Similarity task. **STS** denotes the average of STS12 to STS16.

4 Conclusion & Future Work

We proposed a self-supervised representation learning approach, which leverages the self-contrast of augmented samples obtained by dropout. Despite its simplicity, it achieves comparable results with state-of-the-arts on multiple benchmarks. Future work will deal with sample-specific augmentation to improve the embeddings and, particularly, the representation alignment.

Acknowledgement: We would like to thank Mahdyar Ravanbakhsh for valuable feedback on the manuscript.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task (SemEval 2012)*, pages 385–393.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43.

Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2020. Semantic re-tuning with contrastive tension. In *International Conference on Learning Representations*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A simple framework for contrastive learning of visual representations. pages 1597–1607.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. 2020b. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*.

Xinlei Chen and Kaiming He. 2021. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. 2021. Whitening for self-supervised representation learning. In *International Conference on Machine Learning*, pages 3015–3024. PMLR.

Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. 2020. Cert: Contrastive self-supervised learning for language understanding. *arXiv preprint arXiv:2005.12766*.

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

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 879–895, Online. Association for Computational Linguistics.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. 2020. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *ACM SIGKDD international conference on Knowledge discovery and data mining*.

Tassilo Klein and Moin Nabi. 2020. Contrastive self-supervised learning for commonsense reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7517–7523, Online. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. pages 216–223.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. *arXiv preprint arXiv:1708.00107*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. pages 271–278.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. pages 115–124.

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

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

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

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. pages 1631–1642.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In *the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 200–207.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. Clear: Contrastive learning for sentence representation. *arXiv preprint arXiv:2012.15466*.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. 2021. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*.

Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. 2020. An unsupervised sentence embedding method by mutual information maximization. pages 1601–1610.

Problems with Cosine as a Measure of Embedding Similarity for High Frequency Words

Kaitlyn Zhou[1], **Kawin Ethayarajh**[1], **Dallas Card**[2], and **Dan Jurafsky**[1]

[1]Stanford University, {katezhou, kawin, jurafsky}@stanford.edu
[2]University of Michigan, dalc@umich.edu

Abstract

Cosine similarity of contextual embeddings is used in many NLP tasks (e.g., QA, IR, MT) and metrics (e.g., BERTScore). Here, we uncover systematic ways in which word similarities estimated by cosine over BERT embeddings are understated and trace this effect to training data frequency. We find that relative to human judgements, cosine similarity underestimates the similarity of frequent words with other instances of the same word or other words across contexts, even after controlling for polysemy and other factors. We conjecture that this underestimation of similarity for high frequency words is due to differences in the representational geometry of high and low frequency words and provide a formal argument for the two-dimensional case.

1 Introduction

Measuring semantic similarity plays a critical role in numerous NLP tasks like QA, IR, and MT. Many such metrics are based on the cosine similarity between the contextual embeddings of two words (e.g., BERTScore, MoverScore, BERTR, SemDist; Kim et al., 2021; Zhao et al., 2019; Mathur et al., 2019; Zhang et al., 2020). Here, we demonstrate that cosine similarity when used with BERT embeddings is highly sensitive to training data frequency.

The impact of frequency on accuracy and reliability has mostly been studied on *static* word embeddings like word2vec. Low frequency words have low reliability in neighbor judgements (Hellrich and Hahn, 2016), and yield smaller inner products (Mimno and Thompson, 2017) with higher variance (Ethayarajh et al., 2019a). Frequency also correlates with stability (overlap in nearest neighbors) (Wendlandt et al., 2018), and plays a role in word analogies and bias (Bolukbasi et al., 2016; Caliskan et al., 2017; Zhao et al., 2018; Ethayarajh et al., 2019b). Similar effects have been found in contextual embeddings, particularly for low-frequency senses, which seem to cause difficulties in WSD performance for BERT and RoBERTa (Postma et al., 2016; Blevins and Zettlemoyer, 2020; Gessler and Schneider, 2021). Other works have examined how word frequency impacts the similarity of *sentence* embeddings (Li et al., 2020; Jiang et al., 2022).

While previous work has thus mainly focused on reliability or stability of low frequency words or senses, our work asks: how does frequency impact the semantic similarity of high frequency words?

We find that the cosine of BERT embeddings underestimates the similarity of high frequency words (to other tokens of the same word or to different words) as compared to human judgements. In a series of regression studies, we find that this underestimation persists even after controlling for confounders like polysemy, part-of-speech, and lemma. We conjecture that word frequency induces such distortions via differences in the representational geometry. We introduce new methods for characterizing geometric properties of a word's representation in contextual embedding space, and offer a formal argument for why differences in representational geometry affect cosine similarity measurement in the two-dimensional case.[1]

2 Effect of Frequency on Cosine Similarity

To understand the effect of word frequency on cosine between BERT embeddings (Devlin et al., 2019), we first approximate the training data frequency of each word in the BERT pre-training corpus from a combination of the March 1, 2020 Wikimedia Download and counts from BookCorpus (Zhu et al., 2015; Hartmann and dos Santos, 2018).[2] We then consider two datasets that include

[1] Code for this paper can be found at https://github.com/katezhou/cosine_and_frequency

[2] Additional tools used: https://github.com/IlyaSemenov/wikipedia-word-frequency;

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 401 - 423
May 22-27, 2022 ©2022 Association for Computational Linguistics

pairs of words in context with associated human similarity judgements of words: Word-In-Context (WiC) (expert-judged pairs of sentences with a target lemma used in either the same or different WordNet, Wiktionary, or VerbNet senses) and Stanford Contextualized Word Similarity dataset (SCWS) (non-expert judged pairs of sentences annotated with human ratings of the similarity of two target terms). Using datasets with human similarity scores allows us to account for human perceived similarities when measuring the impact of frequency on cosine (Pilehvar and Camacho-Collados, 2019; Huang et al., 2012).

2.1 Study 1: WiC

Method and Dataset The authors of WiC used coarse sense divisions as proxies for words having the same or different meaning and created 5,428[3] pairs of words in context, labeled as having the same or different meaning:

- same meaning: “I try to avoid the company of gamblers” and “We avoided the ball”
- different meaning: “You must carry your camping gear” and “Sound carries well over water”.

To obtain BERT-based similarity measurements, we use `BERT-base-cased`[4] to embed each example, average the representations of the target word over the last four hidden layers, and compute cosine similarity for the pair of representations.[5]

Relation between frequency and similarity in WiC We want to use ordinary least squares regression to measure the effect of word frequency on the cosine similarity of BERT embeddings. First, we split the WiC dataset into examples that were labeled as having the “same” or “different” meanings. This allows us to to control for perceived similarity of the two words in context — any frequency effects found within these subsets cannot be explained by variation in human judgements. Next, we control for a number of other confounding factors by including them as variables in our OLS regression. For each target lemma we considered:

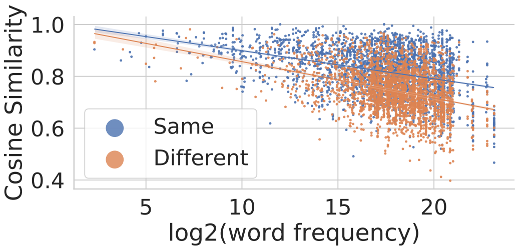


Figure 1: Ordinary Least Squares regression of cosine similarity against frequency, for examples with the same meaning (blue) and different meaning (orange). Both regressions show a significant negative association between cosine similarity and frequency.

frequency: $\log_2$ of the number of occurrences in BERT’s training data

polysemy: $\log_2$ of number of senses in WordNet

is_noun: binary indicator for nouns vs. verbs

same_wordform: binary indicator of having the same wordform in both contexts (e.g., *act/act* vs. *carry/carries*) (case insensitive)

An OLS regression predicting cosine similarity from a single independent factor of $\log_2(\text{freq})$ shows a significant negative association between cosine and frequency among "same meaning" examples (R^2 : 0.13, coeff’s $p < 0.001$) and "different meaning" examples (R^2 : 0.14, coeff’s $p < 0.001$) (see Figure 1). The same negative frequency effect is found across various model specifications (Table 1 in Appendix), which also show significantly greater cosine similarity for those examples with the same wordform, a significant negative association with number of senses, and no difference between nouns and verbs. In summary, we find that using cosine to measure the semantic similarity of words via their BERT embeddings gives systematically smaller similarities the higher the frequency of the word.

Results: Comparing to human similarity To compare cosine similarities to WiC’s binary human judgements (same/different meaning), we followed WiC authors by thresholding cosine values, tuning the threshold on the training set (resulting threshold: 0.8). As found in the original WiC paper, cosine similarity is somewhat predictive of the expert judgements (0.66 dev accuracy, comparable to 0.65 test accuracy from the WiC authors).[6]

Examining the errors as a function of frequency reveals that cosine similarity is a less reliable predictor of human similarity judgements for common

`https://github.com/attardi/wikiextractor`

[3] We used a subset of 5,423 of these examples due to minor spelling differences and availability of frequency data.

[4] `https://huggingface.co/bert-base-cased`

[5] Out-of-vocabulary words are represented as the average of the subword pieces of the word, following Pilehvar and Camacho-Collados (2019) and Blevins and Zettlemoyer (2020); we found that representing OOV words by their first token produced nearly identical results.

[6] The test set is hidden due to an ongoing leaderboard.

terms. Figure 2 shows the average proportion of examples predicted to be the same meaning as a function of frequency, grouped into ten bins, each with the same number of examples. In the highest frequency bin, humans judged 54% of the examples as having the same meaning compared to only 25% as judged by cosine similarity. This suggests that in the WiC dataset, relative to humans, the model underestimates the sense similarity for high frequency words.

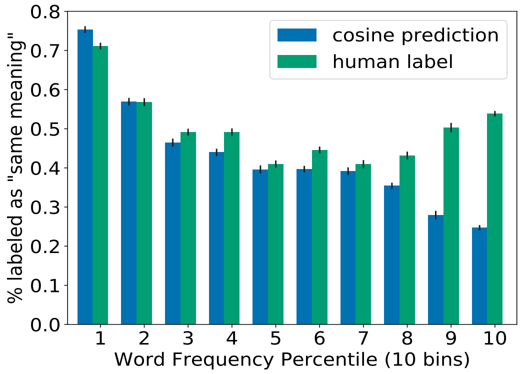


Figure 2: Percentage of examples labeled as having the "same meaning". In high frequency words, cosine similarity-based predictions (blue/left) on average **under**-estimate the similarity of words as compared to human judgements (green/right).

2.2 Study 2: SCWS

Our first study shows that after controlling for sense, cosine will tend to be lower for higher frequency terms. However, the WiC dataset only has binary labels of human judgements, and only indicates similarity between occurrences of the same word. We want to measure if these frequency effects persist across different words and control for more fine-grained human similarity judgements.

Method and Dataset SCWS contains crowd judgements of the similarity of two words in context (scale of 1 to 10). We split the dataset based on whether the target words are the same or different ($break/break$ vs $dance/sing$); this both allows us to confirm our results from WiC and also determine whether frequency-based effects exist in similarity measurements across words.[7] We use the same embedding method as described for WiC, and again use regression to predict cosine similarities from the following features:

frequency: average of $log_2(freq)$ of both words

polysemy: average of $log_2(sense)$ of both words

average rating: average rating of semantic similarity as judged by humans on a scale of 1 to 10 (highest).

Results If we only use frequency, we find that it mildly explains the variance in cosine similarity both within (R^2 : 0.12, coeff's $p < 0.001$) and across words (R^2 : 0.06, coeff's $p < 0.001$). Adding in human average rating as a feature, frequency is still a significant feature with a negative coefficient. High frequency terms thus tend to have lower cosine similarity scores, even after accounting for human judgements. When using all features, the linear regression models explain 34% of the total variance in cosine similarity, with frequency still having a significant negative effect (Table 2 in Appendix). Finally, we verify that for a model with only human ratings, error (true - predicted cosine) is negatively correlated with frequency in held out data (Pearson's $r = -0.18$; $p < 0.01$), indicating an underestimation of cosine in high frequency words (see Figure 5 in Appendix).

This finding suggests that using frequency as a feature might help to better match human judgements of similarity. We test this hypothesis by training regression models to predict human ratings, we find that frequency does have a significant positive effect (Table 3 in Appendix) but the overall improvement over using cosine alone is relatively small ($R^2 = 44.6\%$ vs $R^2 = 44.3\%$ with or without frequency). We conclude that the problem of underestimation in cosine similarity cannot be resolved simply by using a linear correction for frequency.

3 Minimum Bounding Hyperspheres

In order to understand why frequency influences cosine similarity, we analyze the geometry of the contextual embeddings. Unlike static vectors – where each word type is represented by a single point – the variation in contextualized embeddings depends on a word's frequency in training data. We'll call embeddings of a single word type *sibling embeddings* or a *sibling cohort*. To measure variation, we'll use the radius of the smallest hypersphere that contains a set of sibling embeddings (the minimum bounding hypersphere). We tested many ways to measure the space created by high-dimensional vectors. Our results are robust to various other

[7] For consistency across word embeddings, we only use SCWS examples where the keyword appeared lower-cased in context. We reproduced our results with all SCWS examples and found our findings to be qualitatively the same.

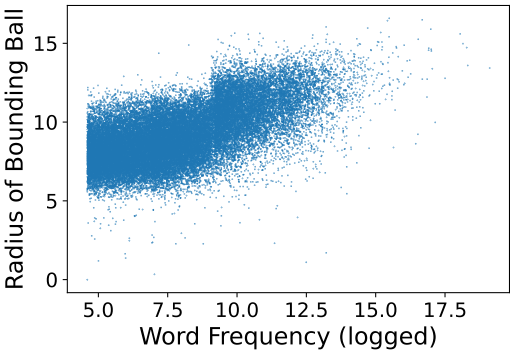


Figure 3: The radius of the minimal bounding ball of sibling embeddings of words is correlated with log(word frequency). (Pearson's $r = 0.62, p < .001$)

measures of variation, including taking the average, max, or variance of pairwise distance between sibling embeddings, the average norm of sibling embeddings, and taking the PCA of these vectors and calculating the convex hull of sibling embeddings in lower dimensions (see Table 29 in the Appendix). Here we relate frequency to spatial variation, providing both empirical evidence and theoretical intuition.

For a sample of 39,621 words, for each word we took 10 instances of its sibling embeddings (example sentences queried from Wikipedia), created contexutalized word embeddings using Hugging Face's `bert-base-cased` model, and calculated the radius of the minimum bounding hypersphere encompassing them.[8] [9] As shown in Figure 3, there is a significant, strong positive correlation between frequency and size of bounding hypersphere (Pearson's $r = 0.62, p < .001$). Notably, since the radius was calculated in 768 dimensions, an increase in radius of 1% results in a hypersphere volume nearly 2084 times larger.[10]

Since frequency and polysemy are highly correlated, we want to measure if frequency is a significant feature for explaining the variance of bounding hyperspheres. Using the unique words of the WiC dataset, we run a series of regressions to predict the radius of bounding hyperspheres. On their own, frequency and polysemy explain for 48% and 45% of the radii's variance. Using both features, frequency and polysemy explains for 58% of the radii's variance and both features are significant – demonstrating that frequency is a significant feature in predicting radii of bounding hyperspheres (Tables 25, 26, 27 in Appendix).

Among the unique words of the WiC dataset, the radii of the target word correlates with training data frequency (Pearson's $r : 0.69, p < 0.001$). Across the WiC dataset, the radii explains for 17% of the variance in cosine similarity (Table 28 in Appendix).[11]

3.1 Theoretical Intuition

Here, we offer some theoretical intuition in 2D for why using cosine similarity to estimate semantic similarity can lead to underestimation (relative to human judgements). Let $\vec{w} \in \mathbb{R}^2$ denote the target word vector, against which we're measuring cosine similarity. Say there were a bounding ball B_x with center $\vec{x_c}$ to which $\vec{w}$ is tangent. If we normalize every point in the bounding ball, it will form an arc on the unit circle. The length of this arc is $2\theta = 2\arcsin\frac{r}{\|x_c\|_2}$:

- Let θ denote the angle made by x_c and the tangent vector $\vec{w}$.
- $\sin\theta = \frac{r}{\|x_c\|_2}$, so the arc length on the unit circle is $r\theta = \arcsin\frac{r}{\|x_c\|_2}$ (normalized points).
- Multiply by 2 to get the arclength between both (normalized) tangent vectors.

Since the arclength is monotonic increasing in r, if the bounding ball were larger—while still being tangent to $\vec{w}$—the arclength will be too.

The cosine similarity between a point in the bounding ball and $\vec{w}$ is equal to the dot product between the projection of the former onto the unit circle (i.e., somewhere on the arc) and the normalized $\vec{w}$. This means that only a certain span of the arclength maps to sibling embeddings $\vec{x_i}$ such that $\cos(\vec{x_i}, \vec{w}) \geq t$, where t is the threshold required to be judged as similar by humans (see Footnote 3 and Figure 4). If B_x were larger while still being tangent to w, the arclength would increase but the span of the arc containing siblings embeddings

[8] Words were binned by frequency and then sampled in order to sample a range of frequencies. As a result, there is a Zipfian effect causing there to be slightly more words in the lower ranges of each bin. We used `https://pypi.org/project/miniball/`

[9] Given the sensitivity of minimum bounding hypersphere to outliers, we'd imagine that frequency-based distortions would be even more pronounced had we chosen to use more instances of sibling embeddings.

[10] the n-dimensional volume of a Euclidean ball of radius R:

$$V_n(R) = \frac{\pi^{n/2}}{\Gamma(\frac{n}{2}+1)}R^n$$

[11] We used 1,253 out of the original 1,265 unique WiC words and 5,412 out of the original 5,428 WiC examples due to availability of frequency data and contextual examples for target words.

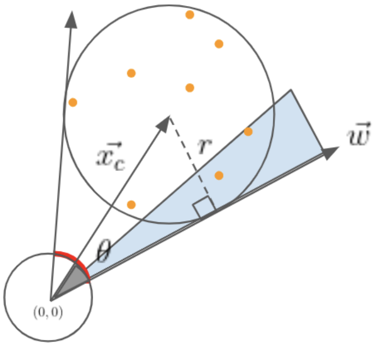


Figure 4: An illustration of how using cosine similarity can underestimate word similarity. The cosine similarity between a contextualized representation (orange) and $\vec{w}$ is the dot product of the former's projection onto the red arc of the unit circle (with length 2θ) and $\hat{w}$. Only points in the blue region are close enough to $\hat{w}$ to be deemed similar by humans. As the bounding ball grows (e.g., with higher frequency words), if it remains tangent to $\vec{w}$, the fraction of points in the blue region will shrink, leading to underestimation.

sufficiently similar to w would not. This means a greater proportion of the sibling embeddings will fail to meet this threshold, assuming that the distribution of sibling embeddings in B_x does not change. Because, in practice, more frequent words have larger bounding balls, depending on how the bounding ball of a word x grows relative to some $\vec{w}$, the similarity of x and w can be underestimated. This helps explain the findings in Figure 2, but it does not explain why more frequent words have lower similarity with themselves across different contexts, since that requires knowledge of the embedding distribution in the bounding ball. The latter is likely due to more frequent words having less anisotropic representations (Ethayarajh, 2019).

4 Discussion and Conclusion

Cosine distance underestimates compared to humans the semantic similarity of frequent words in a variety of settings (expert versus non-expert judged, and within word sense and across words). This finding has large implications for downstream tasks, given that single-point similarity metrics are used in a variety of methods and experiments (Reimers and Gurevych, 2019; Reif et al., 2019; Zhang et al., 2020; Zhao et al., 2019; Mathur et al., 2019; Kim et al., 2021). Word frequency in pre-training data also affects the representational geometry of contextualized embeddings, low frequency words being more concentrated geometrically. One extension of this work might examine how variables such as sentiment and similarity/dissimilarity between sentence contexts could impact both human-judged and embedding-based similarity metrics.

Because training data frequency is something that researchers can control, understanding these distortions is critical to training large language models. Frequency-based interventions might even be able to correct for these systematic underestimations of similarity (e.g., by modifying training data), which could be important where certain words or subjects may be inaccurately represented. For example, Zhou et al. (2022) illustrates how training data frequencies can lead to discrepancies in the representation of countries, and—since frequency is highly correlated with a country's GDP—can perpetuate historic power and wealth inequalities. Future work could also examine how and if frequency effects could be mitigated by post-processing techniques which improve the correlation between human and semantic similarities (Timkey and van Schijndel, 2021).

The semantic similarity distortions caused by the over-and under-representation of topics is another reason why documentation for datasets is critical for increasing transparency and accountability in machine learning models (Gebru et al., 2021; Mitchell et al., 2019; Bender and Friedman, 2018; Ethayarajh and Jurafsky, 2020; Ma et al., 2021). As language models increase in size and training data becomes more challenging to replicate, we recommend that word frequencies and distortions be revealed to users, bringing awareness to the potential inequalities in datasets and the models that are trained on them. In the future, we hope to see research that more critically examines the downstream implications of these findings and various mitigation techniques for such distortions.

Acknowledgements

We sincerely thank Isabel Papadimitriou and our anonymous reviewers for their support, insights, and helpful feedback. This research has been supported in part by a Hoffman-Yee Research Grant from the Stanford Institute for Human-Centered AI, award IIS-2128145 from the NSF, Stanford Data Science, a Stanford Graduate Fellowship, a Facebook Fellowship, and Canada's NSERC.

References

Emily M. Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017.

Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to computer programmer as woman is to homemaker? Debiasing word embeddings. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 4356–4364, Red Hook, NY, USA. Curran Associates Inc.

Aylin Caliskan, Joanna J. Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

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

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019a. Towards understanding linear word analogies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3253–3262, Florence, Italy.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019b. Understanding undesirable word embedding associations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1696–1705, Florence, Italy.

Kawin Ethayarajh and Dan Jurafsky. 2020. Utility is in the eye of the user: A critique of NLP leaderboards. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4846–4853.

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

Luke Gessler and Nathan Schneider. 2021. BERT has uncommon sense: Similarity ranking for word sense BERTology. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*.

Nathan Hartmann and Leandro Borges dos Santos. 2018. NILC at CWI 2018: Exploring feature engineering and feature learning. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 335–340, New Orleans, Louisiana.

Johannes Hellrich and Udo Hahn. 2016. Bad Company—Neighborhoods in neural embedding spaces considered harmful. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2785–2796, Osaka, Japan.

Eric Huang, Richard Socher, Christopher Manning, and Andrew Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 873–882, Jeju Island, Korea.

Ting Jiang, Shaohan Huang, Zihan Zhang, Deqing Wang, Fuzhen Zhuang, Furu Wei, Haizhen Huang, Liangjie Zhang, and Qi Zhang. 2022. Prompt-BERT: Improving BERT sentence embeddings with prompts. *arXiv preprint arXiv:2201.04337*.

Suyoun Kim, Duc Le, Weiyi Zheng, Tarun Singh, Abhinav Arora, Xiaoyu Zhai, Christian Fuegen, Ozlem Kalinli, and Michael L. Seltzer. 2021. Evaluating user perception of speech recognition system quality with semantic distance metric.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130.

Zhiyi Ma, Kawin Ethayarajh, Tristan Thrush, Somya Jain, Ledell Wu, Robin Jia, Christopher Potts, Adina Williams, and Douwe Kiela. 2021. Dynaboard: An evaluation-as-a-service platform for holistic next-generation benchmarking. *Advances in Neural Information Processing Systems*, 34.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2019. Putting evaluation in context: Contextual embeddings improve machine translation evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

David Mimno and Laure Thompson. 2017. The strange geometry of skip-gram with negative sampling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2873–2878, Copenhagen, Denmark.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, page 220–229, New York, NY, USA. Association for Computing Machinery.

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

Marten Postma, Ruben Izquierdo Bevia, and Piek Vossen. 2016. More is not always better: balancing sense distributions for all-words word sense disambiguation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3496–3506, Osaka, Japan. The COLING 2016 Organizing Committee.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of bert. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

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

William Timkey and Marten van Schijndel. 2021. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4527–4546, Online and Punta Cana, Dominican Republic.

Laura Wendlandt, Jonathan K. Kummerfeld, and Rada Mihalcea. 2018. Factors influencing the surprising instability of word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2092–2102, New Orleans, Louisiana.

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

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China.

Kaitlyn Zhou, Kawin Ethayarajh, and Dan Jurafsky. 2022. Richer countries and richer representations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.

A Appendix

For readability, we've summarized the key results from the regressions in 1 and 2. Table 1 contains results from our WiC experiments where we measure frequency's impact on cosine similarity. We control for human judgements of similarity by splitting the dataset by human labels of "same" and "different" meaning words. The same trends hold for the whole dataset as well.

Table 2 contains results from the SCWS experiments we measure frequency's impact on cosine similarity within and across word similarities. Similar to the WiC results, we see that frequency does impact cosine similarity, with higher words having lower similarities.

Table 3 contains results from the SCWS experiments where we measure frequency's impact on human ratings. We see that frequency does not explain human ratings but when used in a model with cosine similarity, frequency has a positive coefficient, indicating it is correcting for the underestimation of cosine similarity.

B Regression results from WiC experiments

Tables 4, 5, 6, 7, 8, 9, 10, 11.

C Regression results from SCWS experiments

Tables 12, 13, 14, 15, 16, 17, 18, 19

D Regression results from SCWS experiments, explaining for the difference between cosine similarity and human judgements

Tables 20, 21, 22, 23, 24.

Cosine similarity is partially predictive of human similarity judgements. The full model shows a significant positive effect of frequency 24 indicating that for a given level of cosine similarity, more frequent terms will judged by humans to be more similar, again demonstrating that cosine under-estimates semantic similarity for frequent terms.

The effect is relatively small, however; for a word that is twice as frequent, the increase in human rating will be 0.0989 (See table 23). Removing frequency from the model reduces R^2 from 40.8% to 40.4%. Polysemy shows the opposite effect; those words with more senses are likely to be rated as less similar. In a model with only cosine and polysemy factors, however, frequency has no relationship with human judgements, indicating that including frequency is correcting for the semantic distortion of cosine in the full model.

E Regression results from minimum bounding hyperspheres

Using frequency and polysemy to explain for the variability in bounding ball radii. Tables 25, 26, 27. Using radius of the bounding ball to explain for the variability of cosine similarity. Table 28.

F Other ways of measuring the space of sibling embeddings

Using a smaller sample of words (10,000 words out of the initial ∼39,000 words), we calculate the space occupied by these sibling embeddings using a variety of other metrics. In each metric, we find strong correlations between (log) frequency and the metric in question (see table 29).

G Residual of Predicted Cosine

For the SCWS dataset, use 1,000 samples as the train set and use the rest as the development set. We train a linear regression model to predict cosine similarity using only human ratings. Taking the difference between cosine similarity and the predicted similarity, we plot this error relative to frequency. We see a negative correlation between this error and frequency $r = -0.18, p < 0.001$, indicating that there is an underestimation of cosine similarity among the high frequency words. Results are shown in Figure 5.

OLS predicting cosine similarity								
WiC	Different Sense Meaning				Same Sense Meaning			
	Model 1	Model 2	Model 3	Model 4	Model 1	Model 2	Model 3	Model 4
$log_2(freq)$	**-0.014**	**-0.012**	**-0.013**	**-0.013**	**-0.011**	**-0.009**	**-0.009**	**-0.010**
$log_2(sense)$	-	**-0.012**	**-0.008**	**-0.009**	-	**-0.006**	**-0.004**	-0.002
same_wordform	-	-	**0.045**	**0.047**	-	-	**0.059**	**0.056**
is_noun	-	-	-	-0.006	-	-	-	**0.008**
R^2	0.127	0.144	0.203	0.204	0.136	0.142	0.241	0.242
Table Number	4	5	6	7	8	9	10	11

Table 1: Coefficients for each of the variables when used in a OLS regression. Bolded numbers are significant. The WiC dataset is split across examples that were rated to have the same or different meaning by experts. Other confounders (polysemy, part-of-speech, word form) were accounted for as features. In model 1, for a word that is twice as frequent, the decrease in cosine similarity will be 0.011.

SCWS	Within Word Examples				Across Words Examples			
	Model 1	Model 2	Model 3	Model 4	Model 1	Model 2	Model 3	Model 4
$log_2(freq)$)	**-0.020**	-	**-0.018**	**-0.016**	**-0.011**	-	**-0.008**	**-0.008**
average rating	-	**0.022**	**0.021**	**0.02**	-	**0.02**	**0.02**	**0.02**
$log_2(sense)$	-	-	-	**-0.019**	-	-	-	-0.001
R^2	0.120	0.225	0.320	0.343	0.059	0.305	0.336	0.337
Table Number	12	13	14	15	16	17	18	19

Table 2: Coefficients for each of the variables when used in a OLS regression. Bolded numbers are significant. The SCWS dataset is split across examples that use the same (within word) or different (across word) target words. Other con-founders (polysemy and average rating) were accounted for as features. In model 1, for a word that is twice as frequent, the decrease in cosine similarity will be 0.02.

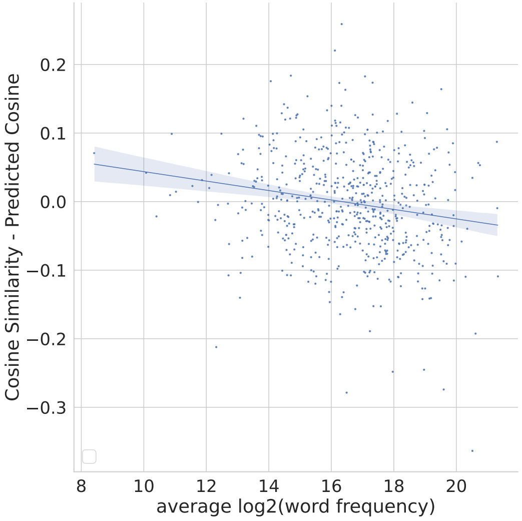


Figure 5: Error in cosine similarity and predicted cosine similarity using human ratings. A negative correlation exists, $r = -0.18, p < 0.001$, indicating an underestimation of cosine similarity among the high frequency words.

OLS Predicting Average Human Rating (Scale of 1 - 10)					
Feature	Model 1	Model 2	Model 3	Model 4	Model 5
avg $log_2(freq)$	-0.057	-	**0.099**	-	**0.076**
avg $log_2(sense)$	-	-	-0.0440	**-0.134**	**-0.189**
cosine	-	**16.345**	**16.665**	**13.513**	**13.809**
same_word	-	-	-	**1.7228**	**1.687**
R^2	0.002	0.404	0.408	0.443	0.446
Table Number	20	21	22	23	24

Table 3: Coefficients for each of the variables when used in a OLS regression. Bolded numbers are significant. Other con-founders (polysemy, same word) were accounted for as features. In model 5, for a word that is twice as frequent, the increase in human rating will be 0.076. Notice that frequency only becomes a significant as a feature when used with cosine, indicating that it is correcting for an underestimation.

Dep. Variable:	Cosine Similarity	**R-squared:**	0.127
Model:	OLS	**Adj. R-squared:**	0.127
Method:	Least Squares	**F-statistic:**	395.1
Date:	Thu, 14 Oct 2021	**Prob (F-statistic):**	3.55e-82
Time:	22:12:38	**Log-Likelihood:**	2947.0
No. Observations:	2713	**AIC:**	-5890.
Df Residuals:	2711	**BIC:**	-5878.
Df Model:	1		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.9976	0.013	77.728	0.000	0.972	1.023
log2(freq)	-0.0141	0.001	-19.876	0.000	-0.015	-0.013

Omnibus:	1.261	**Durbin-Watson:**	1.952
Prob(Omnibus):	0.532	**Jarque-Bera (JB):**	1.189
Skew:	0.044	**Prob(JB):**	0.552
Kurtosis:	3.053	**Cond. No.**	149.

Table 4: OLS regression results predicting cosine similarity among "different meaning" senses.

Dep. Variable:	Cosine Similarity	**R-squared:**	0.144
Model:	OLS	**Adj. R-squared:**	0.144
Method:	Least Squares	**F-statistic:**	228.2
Date:	Thu, 14 Oct 2021	**Prob (F-statistic):**	2.48e-92
Time:	22:12:38	**Log-Likelihood:**	2973.7
No. Observations:	2713	**AIC:**	-5941.
Df Residuals:	2710	**BIC:**	-5924.
Df Model:	2		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.9997	0.013	78.627	0.000	0.975	1.025
log2(freq)	-0.0115	0.001	-14.624	0.000	-0.013	-0.010
log2(senses)	-0.0118	0.002	-7.330	0.000	-0.015	-0.009

Omnibus:	8.024	**Durbin-Watson:**	1.954
Prob(Omnibus):	0.018	**Jarque-Bera (JB):**	9.222
Skew:	0.060	**Prob(JB):**	0.00994
Kurtosis:	3.259	**Cond. No.**	153.

Table 5: OLS regression results predicting cosine similarity among "different meaning" senses.

Dep. Variable:	Cosine Similarity	**R-squared:**	0.203
Model:	OLS	**Adj. R-squared:**	0.202
Method:	Least Squares	**F-statistic:**	230.2
Date:	Thu, 14 Oct 2021	**Prob (F-statistic):**	5.14e-133
Time:	22:12:38	**Log-Likelihood:**	3070.5
No. Observations:	2713	**AIC:**	-6133.
Df Residuals:	2709	**BIC:**	-6109.
Df Model:	3		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.9367	0.013	71.757	0.000	0.911	0.962
log2(freq)	-0.0130	0.001	-16.984	0.000	-0.015	-0.012
log2(senses)	-0.0076	0.002	-4.833	0.000	-0.011	-0.005
same_wordform	0.0447	0.003	14.158	0.000	0.039	0.051

Omnibus:	13.328	**Durbin-Watson:**	1.917
Prob(Omnibus):	0.001	**Jarque-Bera (JB):**	14.587
Skew:	-0.123	**Prob(JB):**	0.000680
Kurtosis:	3.261	**Cond. No.**	163.

Table 6: OLS regression results predicting cosine similarity among "different meaning" senses.

Dep. Variable:	Cosine Similarity	**R-squared:**	0.204
Model:	OLS	**Adj. R-squared:**	0.203
Method:	Least Squares	**F-statistic:**	173.4
Date:	Thu, 14 Oct 2021	**Prob (F-statistic):**	2.26e-132
Time:	22:12:38	**Log-Likelihood:**	3071.8
No. Observations:	2713	**AIC:**	-6134.
Df Residuals:	2708	**BIC:**	-6104.
Df Model:	4		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.9355	0.013	71.569	0.000	0.910	0.961
log2(freq)	-0.0126	0.001	-15.858	0.000	-0.014	-0.011
log2(senses)	-0.0090	0.002	-5.030	0.000	-0.013	-0.005
same_wordform	0.0467	0.003	13.760	0.000	0.040	0.053
is_noun	-0.0061	0.004	-1.629	0.103	-0.013	0.001

Omnibus:	14.009	**Durbin-Watson:**	1.915
Prob(Omnibus):	0.001	**Jarque-Bera (JB):**	15.019
Skew:	-0.135	**Prob(JB):**	0.000548
Kurtosis:	3.244	**Cond. No.**	164.

Table 7: OLS regression results predicting cosine similarity among "different meaning" senses.

Dep. Variable:	Cosine Similarity	**R-squared:**	0.136
Model:	OLS	**Adj. R-squared:**	0.136
Method:	Least Squares	**F-statistic:**	427.3
Date:	Thu, 14 Oct 2021	**Prob (F-statistic):**	2.94e-88
Time:	22:12:38	**Log-Likelihood:**	2926.4
No. Observations:	2710	**AIC:**	-5849.
Df Residuals:	2708	**BIC:**	-5837.
Df Model:	1		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	1.0077	0.009	109.007	0.000	0.990	1.026
log2(freq)	-0.0109	0.001	-20.670	0.000	-0.012	-0.010

Omnibus:	45.476	**Durbin-Watson:**	1.977
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	45.736
Skew:	-0.298	**Prob(JB):**	1.17e-10
Kurtosis:	2.778	**Cond. No.**	103.

Table 8: OLS regression results predicting cosine similarity among "same meaning" senses.

Dep. Variable:	Cosine Similarity	**R-squared:**	0.142
Model:	OLS	**Adj. R-squared:**	0.141
Method:	Least Squares	**F-statistic:**	224.2
Date:	Thu, 14 Oct 2021	**Prob (F-statistic):**	8.17e-91
Time:	22:12:38	**Log-Likelihood:**	2935.6
No. Observations:	2710	**AIC:**	-5865.
Df Residuals:	2707	**BIC:**	-5847.
Df Model:	2		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.9974	0.010	104.755	0.000	0.979	1.016
log2(freq)	-0.0090	0.001	-13.270	0.000	-0.010	-0.008
log2(senses)	-0.0063	0.001	-4.283	0.000	-0.009	-0.003

Omnibus:	38.934	**Durbin-Watson:**	1.973
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	39.612
Skew:	-0.283	**Prob(JB):**	2.50e-09
Kurtosis:	2.823	**Cond. No.**	109.

Table 9: OLS regression results predicting cosine similarity among "same meaning" senses.

Dep. Variable:	Cosine Similarity	**R-squared:**	0.241
Model:	OLS	**Adj. R-squared:**	0.240
Method:	Least Squares	**F-statistic:**	285.7
Date:	Thu, 14 Oct 2021	**Prob (F-statistic):**	4.36e-161
Time:	22:12:38	**Log-Likelihood:**	3100.7
No. Observations:	2710	**AIC:**	-6193.
Df Residuals:	2706	**BIC:**	-6170.
Df Model:	3		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.8928	0.011	84.562	0.000	0.872	0.914
log2(freq)	-0.0092	0.001	-14.435	0.000	-0.010	-0.008
log2(senses)	-0.0035	0.001	-2.513	0.012	-0.006	-0.001
same_wordform	0.0588	0.003	18.728	0.000	0.053	0.065

Omnibus:	80.675	**Durbin-Watson:**	1.981
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	87.234
Skew:	-0.434	**Prob(JB):**	1.14e-19
Kurtosis:	3.139	**Cond. No.**	130.

Table 10: OLS regression results predicting cosine similarity among "same meaning" senses.

Dep. Variable:	Cosine Similarity	**R-squared:**	0.242
Model:	OLS	**Adj. R-squared:**	0.241
Method:	Least Squares	**F-statistic:**	215.8
Date:	Thu, 14 Oct 2021	**Prob (F-statistic):**	6.75e-161
Time:	22:12:38	**Log-Likelihood:**	3103.2
No. Observations:	2710	**AIC:**	-6196.
Df Residuals:	2705	**BIC:**	-6167.
Df Model:	4		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.8952	0.011	84.424	0.000	0.874	0.916
log2(freq)	-0.0096	0.001	-14.547	0.000	-0.011	-0.008
log2(senses)	-0.0022	0.002	-1.457	0.145	-0.005	0.001
same_wordform	0.0560	0.003	16.512	0.000	0.049	0.063
is_noun	0.0078	0.003	2.228	0.026	0.001	0.015

Omnibus:	76.318	**Durbin-Watson:**	1.983
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	82.141
Skew:	-0.421	**Prob(JB):**	1.46e-18
Kurtosis:	3.139	**Cond. No.**	132.

Table 11: OLS regression results predicting cosine similarity among "same meaning" senses.

Dep. Variable:	Cosine Similarity	**R-squared:**	0.120
Model:	OLS	**Adj. R-squared:**	0.115
Method:	Least Squares	**F-statistic:**	28.77
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	2.12e-07
Time:	12:16:53	**Log-Likelihood:**	203.87
No. Observations:	214	**AIC:**	-403.7
Df Residuals:	212	**BIC:**	-397.0
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	1.0762	0.063	17.127	0.000	0.952	1.200
avg_freq	-0.0196	0.004	-5.364	0.000	-0.027	-0.012

Omnibus:	7.823	**Durbin-Watson:**	2.040
Prob(Omnibus):	0.020	**Jarque-Bera (JB):**	9.129
Skew:	-0.307	**Prob(JB):**	0.0104
Kurtosis:	3.804	**Cond. No.**	169.

Table 12: OLS regression results predicting cosine similarity among "same" target words

Dep. Variable:	Cosine Similarity	**R-squared:**	0.225
Model:	OLS	**Adj. R-squared:**	0.221
Method:	Least Squares	**F-statistic:**	61.58
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	2.07e-13
Time:	12:20:20	**Log-Likelihood:**	217.54
No. Observations:	214	**AIC:**	-431.1
Df Residuals:	212	**BIC:**	-424.3
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.5856	0.021	28.308	0.000	0.545	0.626
average_rating	0.0223	0.003	7.847	0.000	0.017	0.028

Omnibus:	31.336	**Durbin-Watson:**	2.183
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	64.374
Skew:	-0.711	**Prob(JB):**	1.05e-14
Kurtosis:	5.279	**Cond. No.**	25.5

Table 13: OLS regression results predicting cosine similarity among "same" target words

Dep. Variable:	Cosine Similarity	**R-squared:**	0.320
Model:	OLS	**Adj. R-squared:**	0.314
Method:	Least Squares	**F-statistic:**	49.70
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	2.06e-18
Time:	12:20:20	**Log-Likelihood:**	231.56
No. Observations:	214	**AIC:**	-457.1
Df Residuals:	211	**BIC:**	-447.0
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.8939	0.060	14.907	0.000	0.776	1.012
avg_freq	-0.0176	0.003	-5.434	0.000	-0.024	-0.011
average_rating	0.0211	0.003	7.893	0.000	0.016	0.026

Omnibus:	18.260	**Durbin-Watson:**	2.246
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	27.332
Skew:	-0.524	**Prob(JB):**	1.16e-06
Kurtosis:	4.402	**Cond. No.**	197.

Table 14: OLS regression results predicting cosine similarity among "same" target words

Dep. Variable:	Cosine Similarity	**R-squared:**	0.343
Model:	OLS	**Adj. R-squared:**	0.334
Method:	Least Squares	**F-statistic:**	36.58
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	4.63e-19
Time:	12:20:20	**Log-Likelihood:**	235.24
No. Observations:	214	**AIC:**	-462.5
Df Residuals:	210	**BIC:**	-449.0
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.9469	0.062	15.214	0.000	0.824	1.070
avg_freq	-0.0161	0.003	-4.983	0.000	-0.022	-0.010
average_rating	0.0198	0.003	7.417	0.000	0.015	0.025
avg_sense	-0.0192	0.007	-2.711	0.007	-0.033	-0.005

Omnibus:	13.882	**Durbin-Watson:**	2.255
Prob(Omnibus):	0.001	**Jarque-Bera (JB):**	18.177
Skew:	-0.458	**Prob(JB):**	0.000113
Kurtosis:	4.095	**Cond. No.**	212.

Table 15: OLS regression results predicting cosine similarity among "same" target words

Dep. Variable:	Cosine Similarity	**R-squared:**	0.059
Model:	OLS	**Adj. R-squared:**	0.058
Method:	Least Squares	**F-statistic:**	87.37
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	3.41e-20
Time:	12:20:20	**Log-Likelihood:**	1557.3
No. Observations:	1406	**AIC:**	-3111.
Df Residuals:	1404	**BIC:**	-3100.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.7858	0.019	42.044	0.000	0.749	0.822
avg_freq	-0.0106	0.001	-9.347	0.000	-0.013	-0.008

Omnibus:	12.804	**Durbin-Watson:**	1.683
Prob(Omnibus):	0.002	**Jarque-Bera (JB):**	16.004
Skew:	-0.130	**Prob(JB):**	0.000335
Kurtosis:	3.453	**Cond. No.**	145.

Table 16: OLS regression results predicting cosine similarity among "different" target words

Dep. Variable:	Cosine Similarity	**R-squared:**	0.305
Model:	OLS	**Adj. R-squared:**	0.304
Method:	Least Squares	**F-statistic:**	614.9
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	7.11e-113
Time:	12:20:20	**Log-Likelihood:**	1770.2
No. Observations:	1406	**AIC:**	-3536.
Df Residuals:	1404	**BIC:**	-3526.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.5366	0.004	150.800	0.000	0.530	0.544
average_rating	0.0208	0.001	24.796	0.000	0.019	0.022

Omnibus:	32.918	**Durbin-Watson:**	1.861
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	39.508
Skew:	-0.302	**Prob(JB):**	2.64e-09
Kurtosis:	3.556	**Cond. No.**	8.58

Table 17: OLS regression results predicting cosine similarity among "different" target words

Dep. Variable:	Cosine Similarity	**R-squared:**	0.336
Model:	OLS	**Adj. R-squared:**	0.335
Method:	Least Squares	**F-statistic:**	355.7
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	1.12e-125
Time:	12:20:20	**Log-Likelihood:**	1803.2
No. Observations:	1406	**AIC:**	-3600.
Df Residuals:	1403	**BIC:**	-3585.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.6684	0.016	40.691	0.000	0.636	0.701
avg_freq	-0.0079	0.001	-8.210	0.000	-0.010	-0.006
average_rating	0.0200	0.001	24.238	0.000	0.018	0.022

Omnibus:	35.771	**Durbin-Watson:**	1.832
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	44.869
Skew:	-0.305	**Prob(JB):**	1.81e-10
Kurtosis:	3.628	**Cond. No.**	156.

Table 18: OLS regression results predicting cosine similarity among "different" target words

Dep. Variable:	Cosine Similarity	**R-squared:**	0.337
Model:	OLS	**Adj. R-squared:**	0.335
Method:	Least Squares	**F-statistic:**	237.1
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	2.09e-124
Time:	12:20:20	**Log-Likelihood:**	1803.4
No. Observations:	1406	**AIC:**	-3599.
Df Residuals:	1402	**BIC:**	-3578.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	0.6670	0.017	40.027	0.000	0.634	0.700
avg_freq	-0.0076	0.001	-7.044	0.000	-0.010	-0.005
average_rating	0.0199	0.001	23.983	0.000	0.018	0.022
avg_sense	-0.0010	0.002	-0.516	0.606	-0.005	0.003

Omnibus:	36.276	**Durbin-Watson:**	1.832
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	45.556
Skew:	-0.308	**Prob(JB):**	1.28e-10
Kurtosis:	3.632	**Cond. No.**	160.

Table 19: OLS regression results predicting cosine similarity among "different" target words

Dep. Variable:	Human Rating	**R-squared:**	0.002
Model:	OLS	**Adj. R-squared:**	0.001
Method:	Least Squares	**F-statistic:**	3.074
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	0.0797
Time:	13:15:45	**Log-Likelihood:**	-3750.9
No. Observations:	1620	**AIC:**	7506.
Df Residuals:	1618	**BIC:**	7517.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	5.0152	0.538	9.330	0.000	3.961	6.070
avg_freq	-0.0568	0.032	-1.753	0.080	-0.120	0.007

Omnibus:	229.333	**Durbin-Watson:**	1.972
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	91.858
Skew:	0.385	**Prob(JB):**	1.13e-20
Kurtosis:	2.124	**Cond. No.**	147.

Table 20: OLS regression results predicting average human ratings.

Dep. Variable:	Human Rating	**R-squared:**	0.404
Model:	OLS	**Adj. R-squared:**	0.403
Method:	Least Squares	**F-statistic:**	1096.
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	6.45e-184
Time:	13:15:45	**Log-Likelihood:**	-3333.6
No. Observations:	1620	**AIC:**	6671.
Df Residuals:	1618	**BIC:**	6682.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	-6.2058	0.314	-19.748	0.000	-6.822	-5.589
cosine_similarity	16.3453	0.494	33.101	0.000	15.377	17.314

Omnibus:	25.721	**Durbin-Watson:**	1.974
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	24.246
Skew:	0.260	**Prob(JB):**	5.43e-06
Kurtosis:	2.703	**Cond. No.**	14.7

Table 21: OLS regression results predicting average human ratings.

Dep. Variable:	Human Rating	**R-squared:**	0.408
Model:	OLS	**Adj. R-squared:**	0.407
Method:	Least Squares	**F-statistic:**	371.8
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	1.31e-183
Time:	13:15:45	**Log-Likelihood:**	-3327.3
No. Observations:	1620	**AIC:**	6663.
Df Residuals:	1616	**BIC:**	6684.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	-7.9168	0.575	-13.778	0.000	-9.044	-6.790
avg_freq	0.0989	0.028	3.473	0.001	0.043	0.155
avg_sense	-0.0440	0.048	-0.911	0.362	-0.139	0.051
cosine_similarity	16.6654	0.500	33.304	0.000	15.684	17.647

Omnibus:	25.797	**Durbin-Watson:**	1.972
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	22.821
Skew:	0.235	**Prob(JB):**	1.11e-05
Kurtosis:	2.657	**Cond. No.**	252.

Table 22: OLS regression results predicting average human ratings.

Dep. Variable:	Human Rating	**R-squared:**	0.443
Model:	OLS	**Adj. R-squared:**	0.442
Method:	Least Squares	**F-statistic:**	428.7
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	7.28e-205
Time:	13:15:45	**Log-Likelihood:**	-3278.2
No. Observations:	1620	**AIC:**	6564.
Df Residuals:	1616	**BIC:**	6586.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	-4.2809	0.379	-11.310	0.000	-5.023	-3.539
avg_sense	-0.1339	0.044	-3.012	0.003	-0.221	-0.047
cosine_similarity	13.5126	0.547	24.707	0.000	12.440	14.585
same_word	1.7228	0.161	10.668	0.000	1.406	2.040

Omnibus:	24.052	**Durbin-Watson:**	2.007
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	20.099
Skew:	0.203	**Prob(JB):**	4.32e-05
Kurtosis:	2.635	**Cond. No.**	46.2

Table 23: OLS regression results predicting average human ratings.

Dep. Variable:	Human Rating	**R-squared:**	0.446
Model:	OLS	**Adj. R-squared:**	0.444
Method:	Least Squares	**F-statistic:**	324.7
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	3.91e-205
Time:	13:15:45	**Log-Likelihood:**	-3274.5
No. Observations:	1620	**AIC:**	6559.
Df Residuals:	1615	**BIC:**	6586.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	-5.5590	0.600	-9.258	0.000	-6.737	-4.381
avg_freq	0.0757	0.028	2.738	0.006	0.021	0.130
avg_sense	-0.1892	0.049	-3.881	0.000	-0.285	-0.094
cosine_similarity	13.8092	0.556	24.816	0.000	12.718	14.901
same_word	1.6872	0.162	10.435	0.000	1.370	2.004

Omnibus:	24.612	**Durbin-Watson:**	2.005
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	19.555
Skew:	0.187	**Prob(JB):**	5.67e-05
Kurtosis:	2.612	**Cond. No.**	285.

Table 24: OLS regression results predicting average human ratings.

Dep. Variable:	Radius of Bounding Ball	**R-squared:**	0.477
Model:	OLS	**Adj. R-squared:**	0.477
Method:	Least Squares	**F-statistic:**	1141.
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	2.96e-178
Time:	15:46:57	**Log-Likelihood:**	-2045.0
No. Observations:	1253	**AIC:**	4094.
Df Residuals:	1251	**BIC:**	4104.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	5.5878	0.187	29.926	0.000	5.221	5.954
log2(freq)	0.3927	0.012	33.774	0.000	0.370	0.416

Omnibus:	15.637	**Durbin-Watson:**	2.053
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	15.928
Skew:	-0.275	**Prob(JB):**	0.000348
Kurtosis:	3.052	**Cond. No.**	86.0

Table 25: OLS regression results predicting radius of bounding ball using frequency

Dep. Variable:	Radius of Bounding Ball	**R-squared:**	0.448
Model:	OLS	**Adj. R-squared:**	0.448
Method:	Least Squares	**F-statistic:**	1015.
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	1.25e-163
Time:	15:46:57	**Log-Likelihood:**	-2078.7
No. Observations:	1253	**AIC:**	4161.
Df Residuals:	1251	**BIC:**	4172.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	9.0630	0.093	97.878	0.000	8.881	9.245
log2(senses)	0.9765	0.031	31.866	0.000	0.916	1.037

Omnibus:	12.796	**Durbin-Watson:**	2.101
Prob(Omnibus):	0.002	**Jarque-Bera (JB):**	13.940
Skew:	-0.193	**Prob(JB):**	0.000940
Kurtosis:	3.344	**Cond. No.**	8.52

Table 26: OLS regression results predicting radius of bounding ball using senses

Dep. Variable:	Radius of Bounding Ball	**R-squared:**	0.583
Model:	OLS	**Adj. R-squared:**	0.582
Method:	Least Squares	**F-statistic:**	872.2
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	7.47e-238
Time:	15:46:57	**Log-Likelihood:**	-1903.7
No. Observations:	1253	**AIC:**	3813.
Df Residuals:	1250	**BIC:**	3829.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
constant	6.0781	0.169	35.937	0.000	5.746	6.410
log2(freq)	0.2581	0.013	20.071	0.000	0.233	0.283
log2(senses)	0.5867	0.033	17.784	0.000	0.522	0.651

Omnibus:	21.564	**Durbin-Watson:**	2.097
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	23.741
Skew:	-0.272	**Prob(JB):**	6.99e-06
Kurtosis:	3.398	**Cond. No.**	88.6

Table 27: OLS regression results predicting radius of bounding ball using frequency and senses

Dep. Variable:	Cosine Similarity	**R-squared:**	0.169
Model:	OLS	**Adj. R-squared:**	0.169
Method:	Least Squares	**F-statistic:**	1103.
Date:	Sat, 12 Mar 2022	**Prob (F-statistic):**	2.51e-220
Time:	15:54:04	**Log-Likelihood:**	5534.8
No. Observations:	5412	**AIC:**	-1.107e+04
Df Residuals:	5410	**BIC:**	-1.105e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
Constant	1.1096	0.010	111.569	0.000	1.090	1.129
Radius of Bounding Ball	-0.0255	0.001	-33.215	0.000	-0.027	-0.024

Omnibus:	1.512	**Durbin-Watson:**	1.721
Prob(Omnibus):	0.470	**Jarque-Bera (JB):**	1.543
Skew:	-0.027	**Prob(JB):**	0.462
Kurtosis:	2.938	**Cond. No.**	109.

Table 28: OLS regression results predicting cosine similarity using radius of the bounding ball.

	Pearson's R	p
Average Pairwise Euclidean Distance	0.601	< 0.001
Max Pairwise Euclidean Distance	0.584	< 0.001
Variance of Pairwise Euclidean Distance	0.292	< 0.001
Average Norm of Embeddings	0.678	< 0.001
Area of convex hull*	0.603	< 0.001

Table 29: Pearson's correlations for numerous other ways of measuring the space occupied by a sibling cohort of ten instances. *To measure the area of a convex hull, we used PCA to projected the embeddings into 2D space and calculated the area. Measuring the convex hull in 768-dimensional space would have required a lot more data (at least 769 samples).

Revisiting the Compositional Generalization Abilities of Neural Sequence Models

Arkil Patel [a] **Satwik Bhattamishra** [b] **Phil Blunsom** [b] **Navin Goyal** [a]

[a] Microsoft Research India
[b] University of Oxford

arkil.patel@gmail.com, navingo@microsoft.com
{satwik.bmishra,phil.blunsom}@cs.ox.ac.uk

Abstract

Compositional generalization is a fundamental trait in humans, allowing us to effortlessly combine known phrases to form novel sentences. Recent works have claimed that standard seq-to-seq models severely lack the ability to compositionally generalize. In this paper, we focus on one-shot primitive generalization as introduced by the popular SCAN benchmark. We demonstrate that modifying the training distribution in simple and intuitive ways enables standard seq-to-seq models to achieve near-perfect generalization performance, thereby showing that their compositional generalization abilities were previously underestimated. We perform detailed empirical analysis of this phenomenon. Our results indicate that the generalization performance of models is highly sensitive to the characteristics of the training data which should be carefully considered while designing such benchmarks in future.

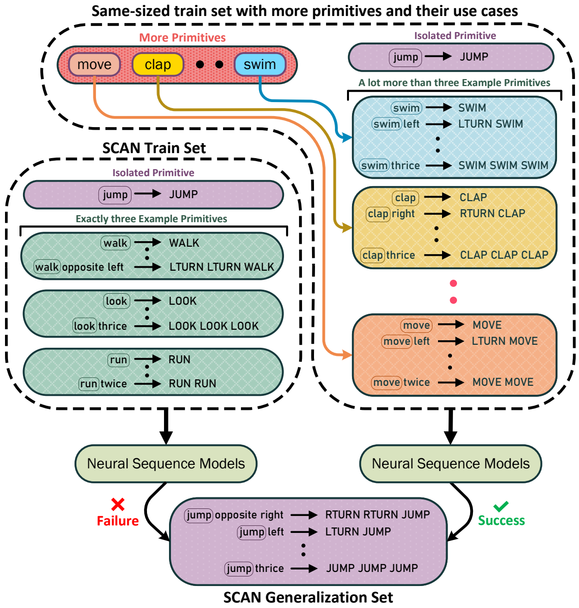


Figure 1: Overview of the SCAN generalization task (left) and our approach (right) that enables standard neural sequence models to generalize compositionally.

1 Introduction

According to the *principle of compositionality*, the meaning of a complex expression (e.g., a sentence) is determined by the meaning of its individual constituents and how they are combined. Humans can effectively recombine known parts to form new sentences that they have never encountered before. Despite the unprecedented achievements of standard seq-to-seq networks such as LSTMs and Transformers in NLP tasks, previous work has suggested that they are severely limited in their ability to generalize compositionally (Lake and Baroni, 2018; Furrer et al., 2020).

Problem Statement. Our work relates to a central challenge posed by compositional generalization datasets such as SCAN (Lake and Baroni, 2018) and Colors (Lake et al., 2019), which we refer to as *one-shot primitive generalization*: The dataset consists of *input-output sentence* pairs (e.g. 'walk twice → WALK WALK'); input sentences are formed from primitive words ('walk') and function words ('twice') and are generated by a context-free grammar (CFG); output sentences are obtained by applying an interpretation function. Crucially, there is a systematic difference between the train and test splits[1]: While the former has a *single* example of an *isolated primitive* (e.g., the primitive definition 'jump → JUMP' in SCAN), the latter consists of compositional sentences with this isolated primitive (e.g. 'jump twice → JUMP JUMP'). See Fig. 1 (left) for an overview of the task.

A model with the right inductive bias should generalize on the test data after having seen compositional expressions with other primitives during training. The need for such inductive bias is justified via psychological experiments (Lake et al., 2019) indicating that humans do have the ability to

[1] We use the term *systematicity* in the rest of the paper to refer to this difference.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 424 - 434
May 22-27, 2022 ©2022 Association for Computational Linguistics

generalize on such tasks. Previous works have suggested that seq-to-seq models lack the appropriate inductive bias necessary to generalize on this task since they achieve near-zero accuracies on both SCAN and Colors benchmarks. This has led to the development of many specialized architectures (Li et al., 2019; Gordon et al., 2020; Chen et al., 2020; Akyurek and Andreas, 2021), learning procedures (Lake, 2019; Conklin et al., 2021) and data augmentation methods (Andreas, 2020; Guo et al., 2020) to solve the task.

Contributions. The primary claim of our paper is that, contrary to prior belief, neural sequence models such as Transformers and RNNs do have an inductive bias[2] to generalize compositionally which can be enabled using the right supervision. **(i)** We show that by making simple and intuitive changes to the training data distribution, standard seq-to-seq models can achieve high generalization performance even with a training set of size less than 20% of the original training set. In particular, if we incorporated examples with more novel primitives in the training set without necessarily increasing the size of the training set (see right part of Fig. 1), then the generalization performance of standard seq-to-seq models improves and reaches near-perfect score after a certain point. Our results also exemplify the importance of the training distribution apart from architectural changes and demonstrate that providing the right supervision can significantly improve the generalization abilities of the models. **(ii)** We investigate the potential cause behind the improvement in generalization performance and observe that the embedding of the isolated primitive becomes more similar to other primitives when the training set has higher number of primitives and their use cases. **(iii)** To understand the phenomenon better, we characterize the effect of different training distributions and model capacities. Our results show that the parameters of the experimental setting play a crucial role while evaluating the generalization abilities of models.

2 Enabling Generalization by Providing the Right Supervision

Setup. We focus on the SCAN and Colors datasets.[3] Both these datasets have exactly one *isolated primitive*. We refer to all other primitives (i.e., those that are also composed with other words to form sentences in the training set) as *example primitives*. Both the SCAN and Colors training sets have exactly three example primitives. The training set of SCAN has 13.2k examples while the test set has 7.7k examples. Colors has just 14 training examples and 8 test examples. More details on implementation and datasets can be found in Appendix A & B. Our source code is available at https://github.com/arkilpatel/Compositional-Generalization-Seq2Seq.

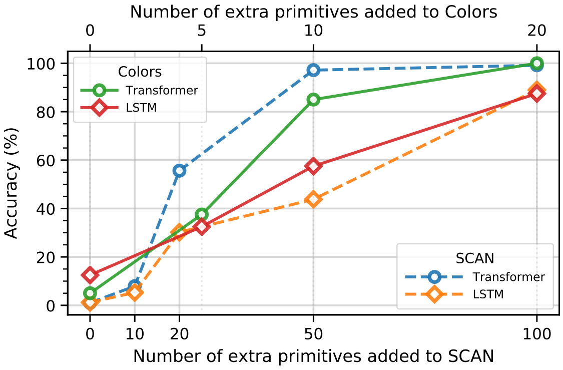


Figure 2: Generalization performance (↑) on SCAN and Colors improves with higher number of example primitives in the training set.

Adding More Primitives. We modify the training set such that the number of distinct example primitives present in the dataset is higher. To do so, we add new primitives to the language which are simply random words (e.g., 'swim', 'clap', etc.) that have the same semantics and follow the same grammar rules as other existing primitives (see Fig. 1 (right) for illustration). These new primitives act as example primitives in our training set. For SCAN, we control the size of the training set such that it is at most the size of the original dataset.[4] To generate the training set, we randomly sample the examples from the new grammar and discard all compositional sentences with the isolated primitive. For each example primitive and the isolated primitive, a primitive definition (such as 'walk → WALK') is also added to the training set. The test set is untouched and remains the same.

Main Observation. Fig. 2 shows the generalization performance of Transformer and LSTM based seq-to-seq models. We observe that there is a clear trend of improvement in compositional gen-

[2] However, note that this inductive bias is not as strong as that of specialized architectures designed for these tasks.

[3] Results on COGS (Kim and Linzen, 2020) can be found in Appendix C.

[4] The training set size $|T|$ is kept fixed by discarding original examples and adding $(|T|/\#primitives)$ examples per primitive. Because of extremely small data size, we cannot do this for Colors while also trying to illustrate our idea.

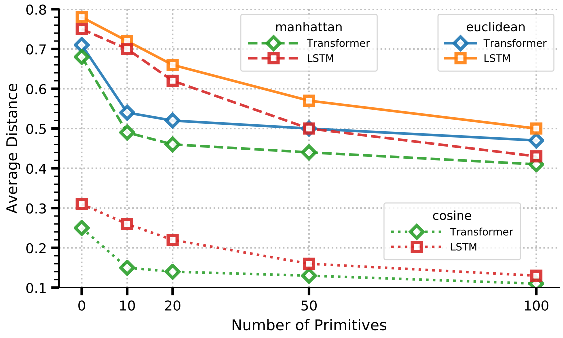


Figure 3: Measuring the distance of embedding of *isolated primitive* with embeddings of example primitives for learned Transformer and LSTM models as we increase the number of example primitives in SCAN.

eralization as we increase the number of example primitives and their use cases. It is surprising to see that on SCAN, Transformers perform on par with some recently proposed specialized architectures (Li et al., 2019; Gordon et al., 2020) and even better than certain architectures (Russin et al., 2019).

Implication. Since the training set still contains only one non-compositional example with the isolated primitive[5] and the test set is untouched, one-shot primitive generalization setting is preserved. Hence our results clearly show that standard neural sequence models have 'some' inductive bias required to generalize on such out-of-distribution tasks even if it is not as strong as that of specialized architectures designed primarily to solve these tasks. Our results are in contradiction to previously suggested limitations of standard seq-to-seq models in terms of primitive generalization (Lake and Baroni, 2018; Furrer et al., 2020; Baroni, 2020). While it is important to develop architectures with better compositional generalization abilities, we wish to highlight that synthetic benchmarks such as SCAN require a model with very strong inductive biases and tend to underestimate the generalization abilities of baseline models.

While we have shown that these models can generalize from one-shot exposure to primitive definitions, our results also hold for the more general case where the one-shot exposure of the primitive is in a sentence (e.g. 'jump twice → JUMP JUMP'). More details regarding these experiments can be found in Appendix D.

Prior Work. Note that our work is unrelated to previous works that propose data augmentation approaches for compositional generalization tasks (Andreas, 2020; Guo et al., 2020; Akyürek et al., 2021). (1) The datasets created by some of these augmentation methods do not preserve the systematic differences between train and test sets, while our datasets do.[6] (2) The objective of these works was to devise a method to improve compositional generalization performance whereas the focus of our work is not to develop a general method; rather we want show that **baseline seq-to-seq models are capable of generalizing compositionally even without breaking systematicity**. (3) These methods add additional data resulting in datasets of larger sizes whereas we control for data size.

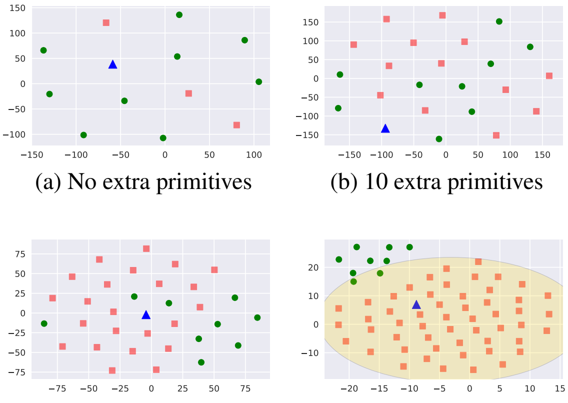

(a) No extra primitives
(b) 10 extra primitives
(c) 20 extra primitives
(d) 50 extra primitives

Figure 4: Visualizing the t-SNE reduced embeddings of isolated primitive (▲), example primitives (■) and non-primitives (●) from a learned Transformer model as we increase number of example primitives in SCAN.

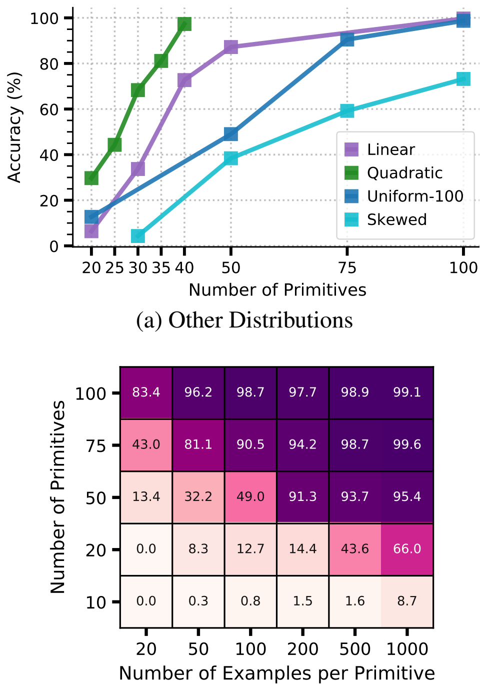
2.1 Analyzing the Embedding of the Isolated Primitive

Our results raise the question: Why do Transformers and LSTMs generalize better when the training data has more example primitives? Compositional generalization in our setting requires a model to learn to apply the same rules to the isolated primitive as it does to the other example primitives. Thus, we analyze the change in the learned embedding of the isolated primitive (such as 'jump') with respect to other primitives in different settings.

In particular, we compare the average distance with other primitives before and after adding certain number of primitives to training data (this is the same setting that was explained earlier in this section). We find that as we increase the number of example primitives in the training set, the em-

[5] Note that our results also hold when there are multiple isolated primitives in the dataset at the same time. This is discussed in Appendix E.5.

[6] We discuss this in more detail in Appendix F.



(a) Other Distributions

Number of Primitives	20	50	100	200	500	1000
100	83.4	96.2	98.7	97.7	98.9	99.1
75	43.0	81.1	90.5	94.2	98.7	99.6
50	13.4	32.2	49.0	91.3	93.7	95.4
20	0.0	8.3	12.7	14.4	43.6	66.0
10	0.0	0.3	0.8	1.5	1.6	8.7

(b) Uniform Distribution

Figure 5: Measuring the generalization performance of Transformer on different types of training set distributions of the SCAN dataset.

bedding of the isolated primitive gets closer to the example primitives (Fig. 3) in terms of Euclidean, Manhattan and Cosine distances. If the embedding of the isolated primitive is closer to the embeddings of the other primitives, then the model is more likely to operate over it in a similar fashion and apply the same rules as it does over the other primitives.

This phenomenon is also illustrated in t-SNE plots (Fig. 4) of the learned embeddings where the embedding of the isolated primitive seems closer to the embeddings of the example primitives when there are more example primitives in the dataset. Hence, a possible reason behind improved generalization performance could be the difference in the learned embeddings.[7] Additional results with the LSTM model and Colors dataset can be found in Appendix E.1.

3 Exploring the Impact of the Parameters of the Experimental Setup

3.1 Impact of Training Distributions

In this section, we analyze the influence of different training distributions on the generalization performance of the model. In the previous experiments, the data generating distribution was uniform over all possible samples. Here, we alter the training data distribution by varying the number of examples for each example primitive. The test set remains unchanged and there will still be only one non-compositional example of the isolated primitive (i.e., the primitive definition) in the training set. We experiment with linearly, quadratically and exponentially increasing probability distribution functions. For instance, in the quadratically increasing case, a training set with 10 example primitives will have one example primitive with 1 compositional example, the next one with 4 compositional examples, another one with 9 compositional examples and so on.[8] Similarly, in the exponentially increasing case (which we also call 'skewed'), 10% example primitives have 500 compositional examples each, 30% have 10 compositional examples each and the remaining have just one compositional example each in the training set. The general idea is that all the example primitives do not have equal representation in the training data. Upon training the models on different distributions, we observed that the models generalize well even with fewer number of example primitives when their distribution is linearly or quadratically increasing (Fig. 5a). On the other hand models struggle to generalize when the distribution is skewed. In that case, most primitives appear in only one or very few compositional sentences in the training data. The failure to generalize on such data implies that extra primitives must be added as part of multiple compositional sentences; just adding the primitive definition or a single example for each example primitive does not help the model to leverage it.

We then try to characterize the relationship between the number of example primitives and the amount of data required for the model to generalize well on the test data, when the example primitives are uniformly distributed. We create different training sets by varying the total number of example primitives, $\#primitives$; for each example primitive, we draw $\#examples$ number of samples uniformly from the CFG. Fig. 5b shows the generalization performance of Transformers for each of these training sets. The size of each training set is the product of the row and column values ($\#primitives \times \#examples$). As expected, the

[7] More fundamental reasons for difference in learned embeddings, such as learning dynamics, are beyond our scope.

[8] In all experimental setups considered in this paper, each example primitive will always have a primitive definition in the training set.

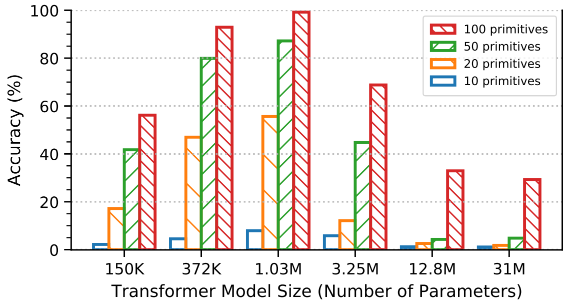


Figure 6: Measuring the generalization performance of a Transformer of varying capacity across increasing number of primitives in the SCAN training set.

upper-right triangle has higher scores indicating that the sample requirement decreases as we add more primitives to the dataset. Surprisingly, the top-left cell indicates that Transformers can achieve high performance even with 2k training examples which is less than **20%** of the original SCAN training set. Additional results with the LSTM model can be found in Appendix E.2.

3.1.1 Understanding Transferability

We wish to check whether the inductive bias that is enabled when a model is trained on more number of example primitives can be transferred to a scenario where the number of example primitives is limited. We create a *pretraining* set with 50 example primitives uniformly distributed, each of them having 200 examples. The *finetuning* set is the original SCAN training set and the test set is the original SCAN test set. The model is first trained from scratch on the pretraining set and then finetuned on the finetuning set.

We find that if we allow all the parameters of the Transformer model to be updated during the finetuning phase on the original SCAN training set, then the model generalizes very poorly. On the other hand, when we freeze the weights of the encoder and decoder after the pretraining phase, and only allow the embedding and output layers to be updated, then the model generalizes near-perfectly on the test set. Our hypothesis is that in the latter setting, the task becomes simpler for the model since it only has to align the embeddings of the newly seen primitives in the finetuning phase with the embeddings of the primitives seen during the pretraining phase. This experiment also indicates that the previously learned rules during pretraining can help a model to compositionally generalize on novel primitives.

3.2 Impact of Model Capacity

We analyze the relationship between the model capacity and the number of example primitives in the training set. We vary the number of primitives as per the description in Section 2. We evaluate the generalization performance of the models while gradually increasing the number of parameters by increasing the size of its embeddings and intermediate representations. For each experiment, we exhaustively finetune the rest of the hyperparameters (e.g., dropout, learning rate, batch size, etc.) to select the best model. Looking at Fig. 6, we observe a general trend in which the model starts to overfit and has poor generalization performance as we increase the model size. Note that all these model configurations are able to achieve near-perfect accuracies on the SCAN random split that does not test for compositional generalization. This shows that carefully controlling the model size is important for achieving compositional generalization. On such small datasets, larger models might simply memorize the input-output mappings in the training set. Indeed, such memorization has been cited as a potential reason to explain why models fail at compositional generalization (Conklin et al., 2021). We also find that as we increase the number of example primitives, the models are less susceptible to overfitting and achieve relatively better generalization performance. Additional results with the LSTM model and Colors dataset can be found in Appendix E.3.

4 Conclusion

While it is essential to make progress in building architectures with better compositional generalization abilities, we showed that the generalization performance of standard seq-to-seq models (often used as baselines) is underestimated. A broader implication of our experiments is that although systematicity must be preserved when designing such benchmarks, it is imperative to carefully explore different parameters associated with the experimental setup to draw robust conclusions about a model's generalization abilities.

Acknowledgements

We thank the anonymous reviewers for their constructive comments. We would also like to thank Kabir Ahuja, Zihuiwen Ye and our colleagues at Microsoft Research India for their valuable feedback and helpful discussions.

References

Ekin Akyürek, Afra Feyza Akyürek, and Jacob Andreas. 2021. Learning to recombine and resample data for compositional generalization. In *International Conference on Learning Representations*.

Ekin Akyurek and Jacob Andreas. 2021. Lexicon learning for few shot sequence modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4934–4946, Online. Association for Computational Linguistics.

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566, Online. Association for Computational Linguistics.

Marco Baroni. 2020. Linguistic generalization and compositionality in modern artificial neural networks. *Philosophical Transactions of the Royal Society B*, 375(1791):20190307.

Paul Bloom. 2000. *How Children Learn the Meanings of Words*. MIT Press.

Xinyun Chen, Chen Liang, Adams Wei Yu, Dawn Song, and Denny Zhou. 2020. Compositional generalization via neural-symbolic stack machines. In *Advances in Neural Information Processing Systems*, volume 33, pages 1690–1701. Curran Associates, Inc.

Henry Conklin, Bailin Wang, Kenny Smith, and Ivan Titov. 2021. Meta-learning to compositionally generalize. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3322–3335, Online. Association for Computational Linguistics.

Róbert Csordás, Kazuki Irie, and Juergen Schmidhuber. 2021. The devil is in the detail: Simple tricks improve systematic generalization of transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 619–634, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daniel Furrer, Marc van Zee, Nathan Scales, and Nathanael Schärli. 2020. Compositional generalization in semantic parsing: Pre-training vs. specialized architectures.

Jonathan Gordon, David Lopez-Paz, Marco Baroni, and Diane Bouchacourt. 2020. Permutation equivariant models for compositional generalization in language. In *International Conference on Learning Representations*.

Demi Guo, Yoon Kim, and Alexander Rush. 2020. Sequence-level mixed sample data augmentation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5547–5552, Online. Association for Computational Linguistics.

Najoung Kim and Tal Linzen. 2020. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9087–9105, Online. Association for Computational Linguistics.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2873–2882. PMLR.

Brenden M Lake. 2019. Compositional generalization through meta sequence-to-sequence learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Brenden M. Lake, Tal Linzen, and Marco Baroni. 2019. Human few-shot learning of compositional instructions.

Yuanpeng Li, Liang Zhao, Jianyu Wang, and Joel Hestness. 2019. Compositional generalization for primitive substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4293–4302, Hong Kong, China. Association for Computational Linguistics.

Santiago Ontañón, Joshua Ainslie, Vaclav Cvicek, and Zachary Fisher. 2021. Making transformers solve compositional tasks.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Jake Russin, Jason Jo, Randall C. O'Reilly, and Yoshua Bengio. 2019. Compositional generalization in a deep seq2seq model by separating syntax and semantics.

Ning Shi, Boxin Wang, Wei Wang, Xiangyu Liu, Rong Zhang, Hui Xue, Xinbing Wang, and Zhouhan Lin. 2021. From scan to real data: Systematic generalization via meaningful learning.

A Implementation Details

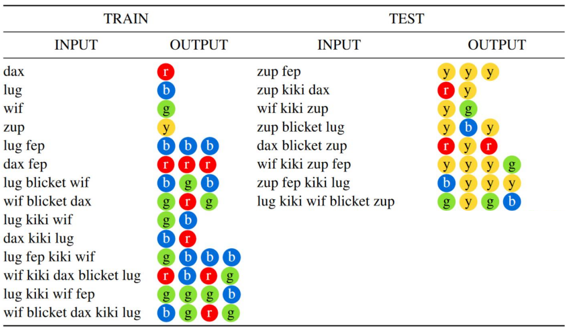
We use 8 NVIDIA Tesla P100 GPUs each with 16 GB memory to run our experiments. All models are implemented in PyTorch (Paszke et al., 2019). We do not use any pretrained models and all embeddings are learnt from scratch. Parameters are updated using Adam optimization. All results are an average of 5 different runs with random seeds. The dataset-specific hyperparameters used for each model are shown in Table 1.

B Primitive Generalization Datasets

In this paper, we show results on three datasets that evaluate primitive generalization.

SCAN (Lake and Baroni, 2018) is a supervised sequence-to-sequence semantic parsing task wherein the natural language input command has to be transformed to the corresponding set of actions. The complete dataset consists of all the commands (a total of 20,910) generated by a phrase-structure grammar and the corresponding sequence of actions, produced according to a semantic interpretation function. The benchmark consists of 4 splits: random, add jump, turn left and length. We work on the 'add jump' split which was designed to test primitive generalization. In this split, the test set (size: 7706) is made up of all the compositional sentences with the primitive 'jump' (which we refer to as the *isolated primitive*). The train set (size: 13,204[9]) has just one example of the isolated primitive (i.e. the primitive definition 'jump → JUMP') and other examples demonstrating the definitions and compositions of the three other primitives (which we refer to as the *example primitives*). Table 2 illustrates the task.

Colors (Lake et al., 2019) is a sequence-to-sequence task that was designed to measure human inductive biases. Apart from the challenge of primitive generalization, this dataset poses an additional challenge of low-resource learning for neural sequence models. The train set has just 14 examples that are either primitive definitions of the four primitives or examples with compositions of the three example primitives and three operations (concatenation, repetition and wrapping). The test set has 8 examples[10] with compositions of the isolated primitive ('zup'). Fig. 7 illustrates the task.



TRAIN		TEST	
INPUT	OUTPUT	INPUT	OUTPUT
dax	r	zup fep	y y y
lug	b	zup kiki dax	r y
wif	g	wif kiki zup	y g
zup	y	zup blicket lug	y b y
lug fep	b b b	dax blicket zup	r y r
dax fep	r r r	wif kiki zup fep	y y y g
lug blicket wif	b g b	zup fep kiki lug	b y y y
wif blicket dax	g r g	lug kiki wif blicket zup	g y g b
lug kiki wif	g b		
dax kiki lug	b r		
lug fep kiki wif	g b b b		
wif kiki dax blicket lug	r b r g		
lug kiki wif fep	g g g b		
wif blicket dax kiki lug	b g r g		

Figure 7: The primitive generalization task in Colors[11]. Note that the test set does not contain the two length generalization examples.

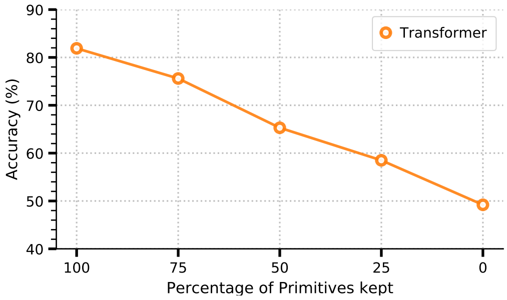


Figure 8: Decrease in generalization performance on our COGS primitive generalization test set with a decrease in the percentage of example primitives and their use cases present in the train set.

COGS (Kim and Linzen, 2020) is a semantic parsing task of mapping English natural language sentences to their corresponding logical forms. Apart from primitive generalization, COGS also evaluates other types of systematic generalization such generalizing to higher depths or generalizing to novel syntactic structures. The size of the train set is 24,155 and that of the test set is 21,000.

C Removing Primitives Hurts Generalization on COGS

Unlike SCAN and Colors, both of which have a single isolated primitive and only 3 example primitives, COGS has 3 isolated primitives - a verb, a common noun and a proper noun which are supported by 80 verbs, 40 common nouns and 20 proper nouns as example primitives. We hypothesize that this high number of example primitives might be one of the reasons behind the high performance of Transformers on COGS (Csordás et al.,

[9] The dataset released by (Lake and Baroni, 2018) is of size 14,670 which has many repetitions of the 'jump → JUMP' primitive definition. In this work, we remove all these repetitions since they do not significantly help in generalization.

[10] The original dataset has two additional examples which evaluate length generalization. Since we focus only on primitive generalization, we do not evaluate on these.

[11] Image taken from Akyurek and Andreas (2021).

	SCAN		COLORS		COGS
Hyperparameters	Transformer	LSTM	Transformer	LSTM	Transformer
Embedding Size	[64, **128**, 256]	[64, **128**, 256]	[16, **32**, 64]	[16, **32**, 64]	[**384**, 512]
Hidden/FFN Size	[**256**, 512]	[**64**, 128]	[16, **32**, 64]	[16, 32, **64**]	[**512**, 1024]
Heads	[**2**, 4]	N/A	[**4**, 8]	N/A	[2, **4**]
Number of Layers	[2, **3**]	[1, **2**]	[**2**, 3]	[1, **2**]	[**2**, 3]
Learning Rate	[3e-4, **5e-4**, 8e-4]	[5e-3, **8e-3**, 1e-2]	[**8e-4**, 1e-3]	[5e-3, **8e-3**, 1e-2]	[3e-4, **5e-4**, 8e-4]
Batch Size	[**128**, 256]	[128, **256**]	[**1**, 2]	[**1**, 2]	[**128**, 256]
Dropout	[**0.1**, 0.2]	[**0.1**, 0.2]	[**0.1**, 0.2]	[**0.1**, 0.2]	[**0.1**, 0.2]
Epochs	150	150	150	150	150
Avg Time/Epoch	30	40	2	3	60

Table 1: Different hyperparameters and the values considered for each of them in the models. The best hyperparameters for each model for all the datasets (with maximum number of primitives of all the settings studied in this paper) are highlighted in bold. Average Time/Epoch is measured in seconds.

TRAIN:	
jump	JUMP
run after run left	LTURN RUN RUN
run	RUN
look left twice	LTURN LOOK LTURN LOOK
TEST:	
jump twice after look	LOOK JUMP JUMP
turn left and **jump**	LTURN JUMP
jump right twice	RTURN JUMP RTURN JUMP

Table 2: An illustration of the primitive generalization task in SCAN.

COMPLEXITY	SENTENCE
1	**jump** twice
2	**jump** thrice and look
3	run twice after **jump** opposite left
4	**jump** around left and walk opposite left twice

Table 3: Sentences of varying complexities featuring the isolated primitive 'jump'.

2021; Ontañón et al., 2021), as far as primitive generalization is concerned.

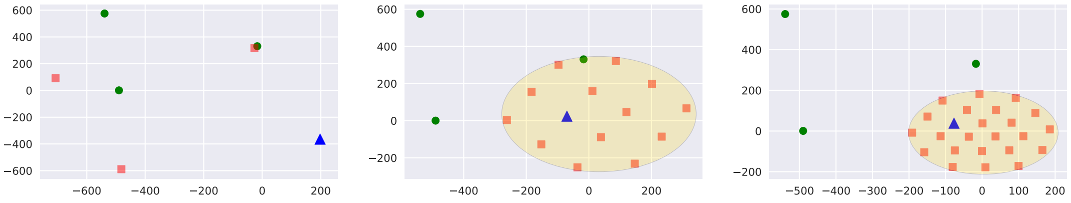
To validate our hypothesis, we systematically reduce the number of example primitives in COGS and evaluate the model. The test set of COGS focusing on primitive generalization consists of 5000 examples. If we directly start removing the primitives from the train set, we risk having out-of-vocabulary tokens in the test set. Hence we select a portion of the test set of size 1218 which exludes 129 example primitives. We will hold this test set fixed and vary the percentage of the 129 example primitives to be inserted in the train set. For each example primitive, samples are drawn uniformly from the original COGS train set. Note that even though the number of example primitives and their use cases will vary in the train set, we control the total train set size to be always 2500 for fair evaluation.

The results of our experiment can be seen in Fig. 8. We see a clear trend of decrease in generalization performance as we decrease the number of example primitives and their use cases. This is in tandem with the results shown in Section 2 and further validates the idea that providing more example primitives and their use cases helps neural sequence models generalize on the primitive generalization task. Our results help explain that the gap in performance of neural sequence models on primitive generalization tasks in COGS and primitive generalization tasks in SCAN or Colors is at least partially caused by the difference in the number of example primitives and their use cases in these datasets.

D Implicit Word Learning

Drawing analogy from human vocabulary acquisition (Bloom, 2000), our primitive generalization setting corresponds to the case when a child is explicitly explained the meaning of a word. But children can learn word meaning from implicit usage. In our setting this would translate to using a primitive in a more complex construction, say 'jump twice → JUMP JUMP' instead of the original 'jump → JUMP'. It would be interesting to evaluate how well seq-to-seq models learn the meanings of words from a single sentence and whether they learn to use that word compositionally with other words.

We consider the 'add jump' split in SCAN. Instead of providing the 'jump → JUMP' primitive definition in the train set, we provide one compositional sentence featuring 'jump'. We vary the complexity of this sentence as shown in Table 3. Similar to the case of providing only the primitive definition, we observe that models are unable to



(a) No extra primitives (b) 10 extra primitives (c) 20 extra primitives

Figure 9: Visualizing the t-SNE reduced embeddings of isolated primitive (▲), example primitives (■) and non-primitives (●) from a learned LSTM model as we increase the number of example primitives in the Colors train set.

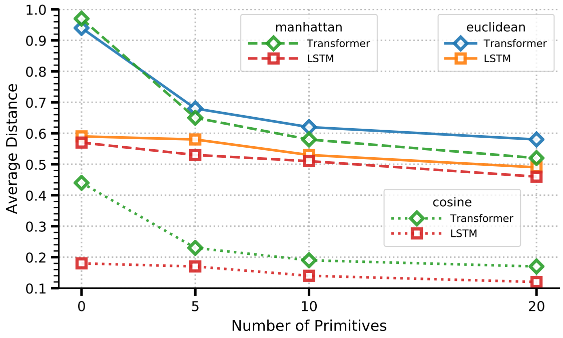
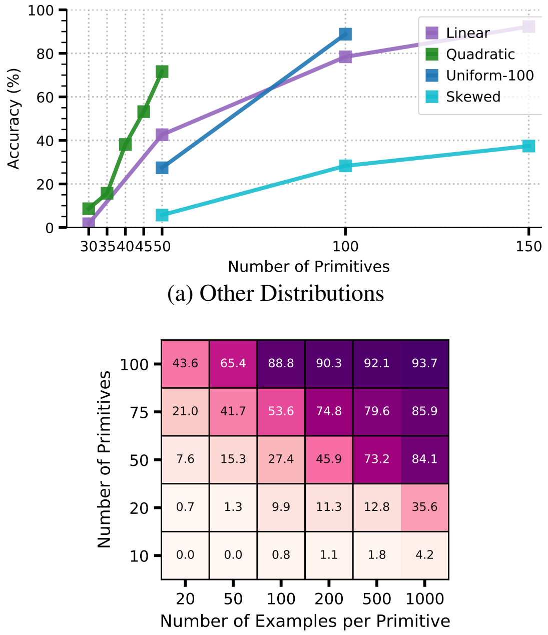


Figure 10: Measuring the similarity of the embedding of *isolated primitive* with the embeddings of example primitives for learned Transformer and LSTM models as we increase the number of example primitives in the Colors train set.



(a) Other Distributions

(b) Uniform Distribution

Figure 11: Measuring the generalization performance of LSTM on different types of train set distributions of the SCAN dataset.

generalize and achieve near-zero accuracies.

We now wish to see whether the presence of more number of primitives and their sentences in the train set helps a model generalize in this scenario (like it did for primitive definitions as shown in Section 2). We consider the setup of having 100 primitives and their sentences in the train set (Section 2) apart from the one compositional sentence with the word 'jump'. We find that models are able to achieve near-perfect generalization accuracies.

This shows that our idea holds more generally: Adding more primitives and their sentences helps a model effectively learn the meaning of a new primitive, whether specified explicitly via a primitive definition or implicitly in a sentence.

E Details of Experimental Setups and Other Results

E.1 Embedding of Isolated Primitive

We scale the embedding vectors to unit $L2$-norm for calculating the euclidean distance and unit $L1$-norm for calculating the manhattan distance. For Colors dataset as well, we compare the average distance with other primitives before and after adding primitives to the training data. We again find that as we increase the number of example primitives in the training set, the embedding of the isolated primitive ('zup') gets closer to the example primitives (refer to Fig. 10) in terms of Euclidean, Manhattan and Cosine Distances.

We additionally show the t-SNE plots of the learned embeddings for the LSTM model on the Colors dataset (Fig. 9).

E.2 Impact of Training Distributions

In Section 3.1, we showed results of the Transformer model on various train set distributions of the SCAN dataset. We also experimented with the LSTM model, the results of which can be found in Fig. 11. We see the same trend as we saw for Transformers.

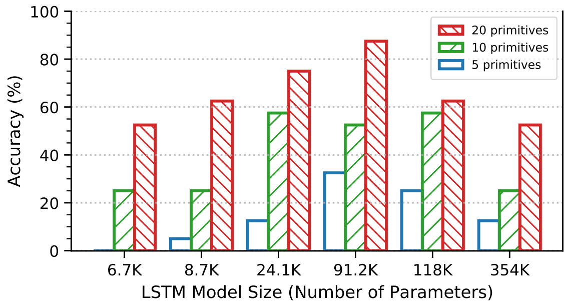


Figure 12: Measuring the generalization performance of an LSTM of varying capacity across increasing number of primitives in the Colors train set.

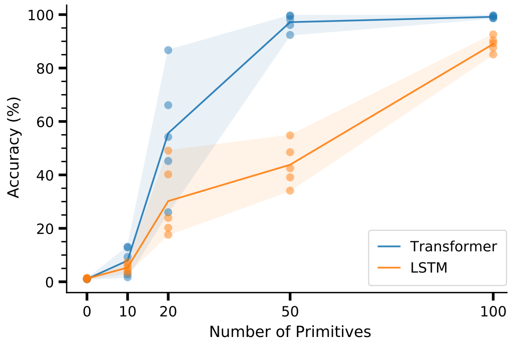


Figure 13: Generalization performance on SCAN across different runs with random seeds.

E.3 Impact of Model Capacity

In Section 3.2, we showed results of varying sizes of Transformers trained on datasets with different number of example primitives. We also experimented with the LSTM model, the results of which on the Colors dataset can be found in Fig. 12. We see the same trend as we saw for Transformers.

E.4 Variance Across Different Runs

We plot the generalization accuracies of the Transformer and LSTM models on SCAN and Colors datasets over 5 different runs with random seeds in Fig. 13-14. Both models displayed a high degree of variance in generalization performance on both datasets. It is interesting to see that the variance decreases with increasing number of primitives.

E.5 Evaluation on Multiple Isolated Primitives

Our results are valid not just when there is a single isolated primitive, but even when there are multiple isolated primitives that are used compositionally at test time. While we believe that this holds trivially due to the symmetry of the setup, for completeness, we provide empirical evidence. We consider the setting on SCAN in which the train set has a total of

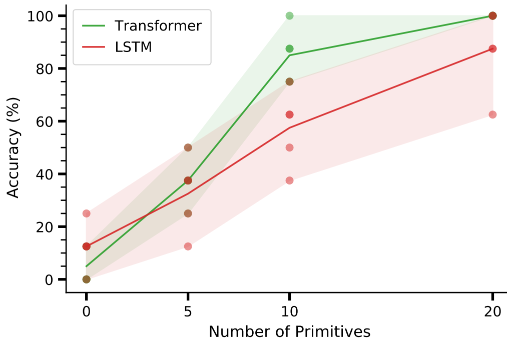


Figure 14: Generalization performance on Colors across different runs with random seeds.

100 example primitives uniformly distributed. To this train set, in addition to the primitive definition of 'jump' (i.e., 'jump → JUMP'), we add 9 other primitive definitions of newly introduced isolated primitives. Thus, while the size of the train set in this setting was 13185, the size of the new train set is 13194. We then extract templates from the original SCAN test set and exhaustively populate these templates with the 10 isolated primitives. Hence, while the size of the original test set was 7706, the size of the new test set is 77060.

We evaluated Transformers on this data. The best model achieved 94.5% accuracy on the complete test set, thereby showing that our methodology and results are valid even when there are multiple isolated primitives in the dataset at the same time.

F A Note on Other Data Augmentation Methods

Applying data augmentation methods such as GECA (Andreas, 2020) on SCAN will lead to addition of training examples in which the input sentences are compositions of the isolated primitive 'jump'. This breaks the systematicity of the setup. While such automatic data augmentation approaches are important resources for enabling compositional generalization, a model that performs well on this modified split cannot be considered to be able to generalize compositionally.

Shi et al. (2021) proposed a data augmentation method based on the theory of meaningful learning. Similar to our work, they also augment the train set by adding more primitives (e.g. 'jump_0', 'jump_1', ..., 'jump_n'). However, compared to our work, their setup is completely different: The new primitives that they add to the train set are all still mapped to the output token of an example primitive ‘jump’, which is ‘JUMP’ (i.e. ‘jump_0 → JUMP’, ..., ‘jump_n → JUMP’). Their train set has examples showing compositions of ‘jump’ while their test set evaluates for novel compositions of the newly added primitives. We argue that their setup cannot be considered one-shot primitive generalization since now the model can see the output token ‘JUMP’ in composition with other words. We claim that this familiarity with the output token enables a model to generalize well on the test data even if the newly added primitives are only presented one-shot in the train set. Indeed, Lake and Baroni (2018) also suggested that the reason why models are able to do well on the ‘turn left’ split of SCAN is because the train set consists of many examples that have the output token ‘LTURN’ used compositionally.

To validate our claim, we propose a simple experiment. In the original SCAN ‘add jump’ split, we map ‘jump → WALK’ instead of ‘jump → JUMP’ for all examples (primitive definition as well as compositional sentences) in both the train and test sets. In this setup, even though the input word ‘jump’ is seen only once at train time, it’s mapping ‘WALK’ is used compositionally in many examples. On evaluating a Transformer model on this split, we found that it achieves a near-perfect accuracy. This shows that providing compositional examples with the output token of the isolated primitive not only breaks systematicity, but is the reason behind the high performance of models in that setting.

A Copy-Augmented Generative Model for Open-Domain Question Answering

Shuang Liu[1]*, **Dong Wang**[2], **Xiaoguang Li**[1], **Minghui Huang**[2], **Meizhen Ding**[2]
[1] Huawei Noah's Ark Lab
[2] AI Application Research Center (AARC)
Huawei Technologies Co., Ltd
{liushuang30, wangdong153}@huawei.com

Abstract

Open-domain question answering is a challenging task with a wide variety of practical applications. Existing modern approaches mostly follow a standard two-stage paradigm: retriever then reader. In this article, we focus on improving the effectiveness of the reader module and propose a novel copy-augmented generative approach that integrates the merits of both extractive and generative readers. In particular, our model is built upon the powerful generative model FiD (Izacard and Grave, 2021b). We enhance the original generative reader by incorporating a pointer network to encourage the model to directly copy words from the retrieved passages. We conduct experiments on the two benchmark datasets, NaturalQuestions and TriviaQA, and the empirical results demonstrate the performance gains of our proposed approach.

1 Introduction

Open-domain question answering (ODQA) focuses on providing highly precise answers to natural language questions from a large collection of unstructured text data (Voorhees, 1999). With the pioneering work of DrQA (Chen et al., 2017), modern approaches to ODQA commonly adopt a simple two-stage *retriever-reader* pipeline, that firstly retrieve a relatively small number of support passages (Karpukhin et al., 2020; Min et al., 2021b; Yamada et al., 2021), followed by the reader identifying the answer.

The reader models can be broadly categorized into two classes: extractive (Chen et al., 2017; Asai et al., 2020; Karpukhin et al., 2020) and generative (Izacard and Grave, 2021b; Lewis et al., 2020b; Wu et al., 2021). Recently, benefiting from the powerful ability of large-scale pre-trained encoder-decoder language models (Lewis et al., 2020a; Raffel et al., 2019) and the capability of aggregating information from multiple passages (Izacard and Grave, 2021b), generative approaches have achieved in general better performance than extractive methods.

Question: where was a hologram for the king filmed?
Passages (Truncated): title: A Hologram for the King (film) context: Production was set to begin in first quarter of 2014. Principal photography commenced on March 6, 2014 in Morocco. Filming also took place in Hurghada in Egypt, as well as in Berlin and Düsseldorf in Germany. Shooting wrapped in June 2014.
Answer: Hurghada in Egypt, Berlin and Düsseldorf in Germany
FiD: Dubai in Germany
FiD-PGN: Hurghada in Egypt

Question: who has the most trophies in la liga?
Passages (Truncated): title: La Liga context: A total of 62 teams have competed in La Liga since its inception. Nine teams have been crowned champions, with Real Madrid winning the title a record 33 times and Barcelona 25 times.
Answer: Real Madrid
FiD: 33
FiD-PGN: Real Madrid

Table 1: Comparisons of answers generated by FiD and our approach. The orange text represents supportive sentences.

Compared to extractive models, generative models generate text more freely, which makes it often suffer from the problem of producing hallucinated text that is factual inaccuracy or inconsistent to the input. This problem has been addressed in tasks like text summarization (Maynez et al., 2020) and machine translation (Zhou et al., 2021). We found that the phenomenon also happens in ODQA. As shown in Table 1, the answer "Dubai in Germany" produced by the generative model FiD (Izacard and Grave, 2021b) is factual incorrect and the answer "33" in the second example is not coherent to the question. While in both cases, the ground-truth answers are present in the retrieved passages. Thus, we hypothesize that if we could put a constraint on the produced words to the input text, the generated answer will be more faithful.

Inspired by the work of See et al. (2017), we enhance the generative model with a pointer net-

[*] This work was done when she was at AARC.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 435 - 441
May 22-27, 2022

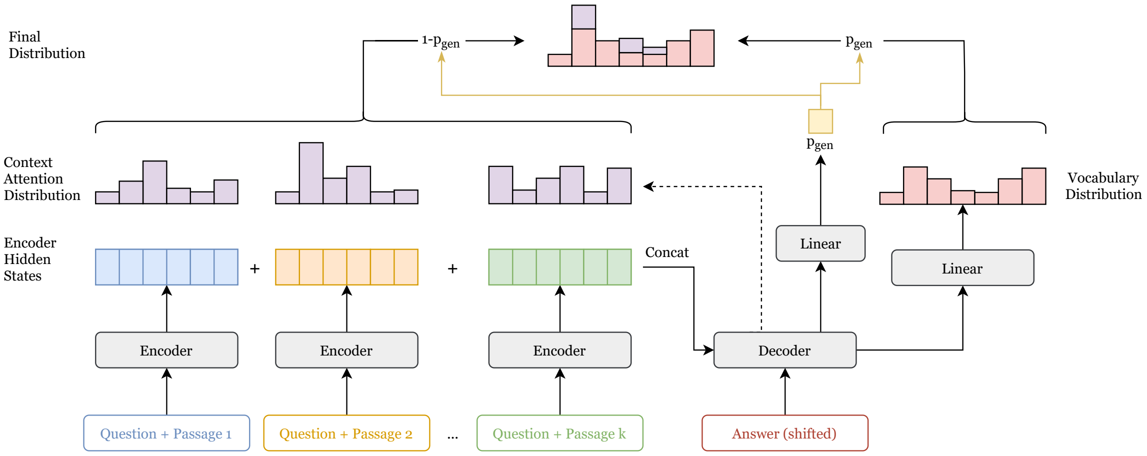


Figure 1: The overall architecture of our proposed model. We add a linear layer to calculate the generation probability, which decides the weights of generating words from vocabulary or copying from source passages.

work (Vinyals et al., 2015), that enables the model to directly copy text from the retrieved passages while retains the ability of generating new words when the true answers are not explicitly present in the input. To be more specific, our model fusion-in-decoder pointer-generator network (FiD-PGN) is built upon the state-of-the-art model FiD. We reuse the encoder-decoder attention scores as the copy distribution to reduce the computational cost. Compared to FiD, we achieve comparative or even better accuracy on the NaturalQuestions (NQ) (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017) benchmarks, with less passages used in training. Our experiments results show the effectiveness and efficiency of our model.

2 Related Work

2.1 Open-Domain Question Answering

In this era of data explosion, ODQA offers a way to rapidly and accurately fulfill user's information needs, and hence has recently received significant attention from both industry and academia (Min et al., 2021a). Following the work of DrQA (Chen et al., 2017), most recent works build a two-stage *retriever-reader* system to tackle the problem. The retriever aims at retrieving supportive passages to the given question from a large document corpus. The reader intends to find answer of the question from the first stage retrieved passages. Early work of Chen et al. (2017) adapts a BiLSTM architecture with various lexical and semantic features from the question and passages as inputs. Later, with the emergence of large-scale pre-trained language models, readers based on pre-trained models such as BERT (Devlin et al., 2019) and T5 (Raffel et al., 2019) have become a common approach (Yang et al., 2019; Izacard and Grave, 2021b; Karpukhin et al., 2020).

2.2 Generative Readers

Compared to extractive models which extract spans from the retrieved passages, generative models are able to produce new words out of the retrieved passages, and thus provide a more flexible modeling framework. Min et al. (2020) and Lewis et al. (2020b) concatenate the given question with top retrieved passages and feed the concatenation to the BART model (Lewis et al., 2020a). Izacard and Grave (2021b) separately encodes the question with each top retrieved passage, then takes the concatenation of the encoder outputs as input to the decoder. Their method provide a way to better aggregate evidence from multiple passages and improve the performance significantly. FiD-KD (Izacard and Grave, 2021a) is an extension of FiD model that increases the accuracy of passage retrieval by training the dense retriever with the guidance of the FiD reader iteratively.

2.3 Pointer-Generator Network

Pointer-Generator Network (See et al., 2017) is an extension of the sequence-to-sequence model by integrating a copy mechanism (Vinyals et al., 2015) into the generator. At each decoding stage, the model is able to either directly copy a word from the input or generate one with certain probability, and thus can be viewed as a combination of extractive and generative approaches. It has been frequently used in natural language tasks like summarization (Gu et al., 2016; See et al., 2017; Gehrmann et al., 2018) and neural machine translation (Luong et al., 2015; Gu et al., 2018), but its application to ODQA has been less explored.

3 Method

Our model follows the standard two-stage *retriever-reader* framework with a focus on the enhancement of the reader module built upon the FiD reader. We adopt the retriever results of FiD-KD, where a dense retriever similar to DPR (Karpukhin et al., 2020) is used. A pointer network is integrated into the FiD reader to facilitate copying words from the retrieved passages. The overall reader architecture is depicted in Figure 1.

Reader Encoder. The reader encoder of our model is identical to the one of FiD reader. We firstly concatenate the given question q with each retrieved passage p_i as $x_i = [q; p_i]$. Next, we pass each x_i individually to the reader encoder, i.e., the encoder of T5 or BART model, and obtain the hidden representations $h_i = (h_{i,1}, h_{i,2}, \ldots, h_{i,n})$ of the question-passage pair where $h_{i,j} \in \mathbb{R}^d$ and d is the model dimension. Finally, we concatenate all the hidden representations of top-k passages $\{h_1, \ldots, h_k\}$ as input to the decoder.

Reader Decoder. Our approach mainly differs from FiD reader in the decoder module by adding a pointer network. Specifically, at each decoding step t, let $e_t \in \mathbb{R}^d$ be the embedding vector of the input token at this step, and denote $s_t^L \in \mathbb{R}^d$ as the output representation of the last layer L of transformer decoder, then the probability of generation is given as follows,

$$p_{\text{gen}} = \sigma(w_e^T e_t + w_s^T s_t^L + b) \qquad (1)$$

where $w_e \in \mathbb{R}^d$, $w_s \in \mathbb{R}^d$ and $b \in \mathbb{R}$ are all learnable parameters and $\sigma(\cdot)$ represents the sigmoid function. In addition, the probability of copying is $1 - p_{\text{gen}}$.

Next, let $\mathcal{V}$ denote the vocabulary containing words for the generative model and $|\mathcal{V}|$ be the size of the vocabulary. Then at step t, the probability distribution of words generation over the vocabulary is computed as,

$$P_{\text{vocab}} = \text{softmax}(W_E s_t^L) \qquad (2)$$

where $W_E \in \mathbb{R}^{|V| \times d}$ is a learnable weight matrix.

Benefiting from the encoder-decoder attention layer in transformer architecture, we directly utilize the cross-attention score α_t^L of the last decoder layer L over the source tokens for the target token y_t as copy distribution. Then the probability of selecting y_t in source sequence is calculated as,

$$P_{\text{ctx}}(y_t) = \sum\nolimits_{j: x_{1:k,j} = y_t} \alpha_{t,j}^L \qquad (3)$$

where $x_{1:k}$ denotes the concatenation of the top-k retrieved passages, $x_{1:k,j}$ is the j-th token of $x_{1:k}$, and $\alpha_{t,j}^L$ is the j-th element of α_t^L. If y_t is not present in the top-k retrieved passages, $P_{\text{ctx}}(y_t)$ will be zero.

Finally, put all the above together, the target token y_t could both be generated from vocabulary with probability p_{gen}, and copy from the source passages. The final prediction probability is defined as

$$P(y_t) = p_{\text{gen}} P_{\text{vocab}}(y_t) + (1 - p_{\text{gen}}) P_{\text{ctx}}(y_t). \quad (4)$$

4 Experiments

4.1 Datasets

We evaluate the performance of our approach on two standard ODQA datasets, NQ and TriviaQA. The NQ dataset comprises real queries that user issued on Google search engine along with answers. The TriviaQA dataset consists of question-answer pairs collected from trivia and quiz-league websites. The details of data statistics are listed in Table 2. It can be seen that TriviaQA has on average longer question length than NQ, indicating that questions in TriviaQA are relatively more complex. We use the data released on the repository of FiD[1], containing question-answer pairs and top-100 passages retrieved by FiD-KD.

Statistics	NQ	TriviaQA
Train	79,168	78,785
Validation	8,757	8,837
Test	3,610	11,313
Avg. Qlen	9.3	16.9
Avg. Alen	2.4	2.2

Table 2: Summary statistics of the two datasets. Avg. Qlen and Avg. Alen denote the average number of tokens per question and answer, respectively.

[1] https://github.com/facebookresearch/FiD

Model	Reader Size	Top-k	NQ	TriviaQA
DPR (BERT-base) (Karpukhin et al., 2020)	110M	24	41.5	57.9
RAG-Seq (BART-large) (Lewis et al., 2020b)	406M	50	44.5	56.8
FiD (T5-base) (Izacard and Grave, 2021b)	220M	100	48.2	65.0
FiD-KD (T5-base) (Izacard and Grave, 2021a)	220M	100	49.6	**68.8**
FiD-KD (Our implementation)	220M	25	48.5	67.5
FiD-PGN	220M	25	**51.4**	68.4

Table 3: Exact match (EM) scores on NQ and TriviaQA test sets. Top-k indicates the number of retrieved passages used during reader training. The performance of SOTA model is in **bold** and the second best model is in underline.

4.2 Implementation Details

We follow the experimental settings as in FiD. Our model is initialized with a pre-trained T5-base model, and trained using AdamW (Loshchilov and Hutter, 2017) algorithm with a learning rate of 10^{-4}, linear scheduling with 15k total steps and 1k warm-up steps. Moreover, we train our model using the top-25 retrieved passages for each question and set the batch size as 64 due to computational limitation. All experiments are run on eight Nvidia V100 32GB GPUs.

4.3 Results

Table 3 shows the experimental results of our model and other approaches on the test sets, evaluated with the standard exact match (EM) score (Rajpurkar et al., 2016). For a fair comparison, we retrained the FiD reader on the top-25 retrieved passages to match our experimental settings.

As shown in Table 3, our model outperforms FiD-KD on both NQ and TriviaQA datasets under the same setting. This demonstrates that the pointer network could help to generate answers more accurately. It is worth noting that, compared with FiD-KD trained with the top-100 retrieved passages, our model achieves comparative or even better results with only 1/4 of the input data and without introducing many parameters (only 1537 extra parameters are added), indicating the efficiency of our model.

5 Analysis

Generation Probability. We explore the probability of generation during training to further investigate the effects of the pointer module. As shown in Figure 2, the generation probability p_{gen} in TriviaQA is always higher than the one in NQ. Note that a higher generation probability means that more tokens are produced from the vocabulary instead of copying from the input. We conjecture that this phenomenon is caused by the different question types. As stated in Rogers et al. (2021), Trivia questions are more like probing questions. Compared to the information-seeking questions in NQ, probing questions tend to need more complex reasoning, and thus it is difficult to directly extract relevant tokens from input texts. Moreover, this observation is also consistent with the results that the improvements of our model over FiD reader is smaller in TriviaQA than the one in NQ (0.9 vs. 2.9 EM for TriviaQA and NQ, respectively).

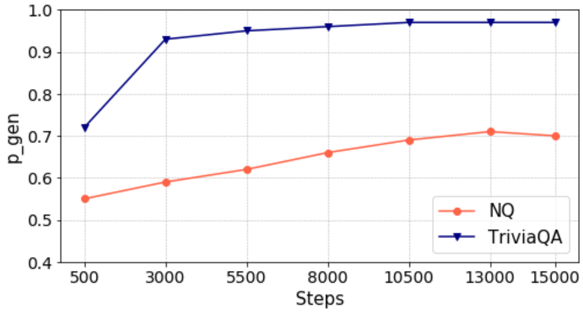


Figure 2: Generation probability p_{gen} over training steps on NQ and TriviaQA.

Test-Train Overlap Evaluation. The study of test-train overlap (Lewis et al., 2021) provides valuable insights into the model's question answering behavior. We evaluate our model on the same test data splits as in Lewis et al. (2021). Table 4 reports the results with respect to three kinds of test-train overlaps. It can be seen that our approach improves most over FiD reader on "No Overlap" category, the most challenging setting, indicating a better generalization ability to question answering.

Training with Varying Number of Passages. Figure 3 shows the performance of our model and FiD reader with regard to different number of retrieved training passages. We train both models with top-k passages ($k \in \{1, 5, 10, 25\}$) and evaluate on the development sets with the same number of pas-

Dataset	Overlap Type	FiD	FiD-PGN	Δ
NQ	Total	48.5	**51.4**	2.9
	Question Overlap	73.5	**75.9**	2.4
	Answer Overlap Only	41.0	**45.1**	4.1
	No Overlap	28.8	**38.4**	9.6
TriviaQA	Total	67.5	**68.4**	0.9
	Question Overlap	88.4	**89.6**	1.2
	Answer Overlap Only	66.9	**68.4**	1.5
	No Overlap	41.5	**43.4**	1.9

Table 4: Test-train overlap evaluation on NQ and TriviaQA test sets. Exact match (EM) scores are reported.

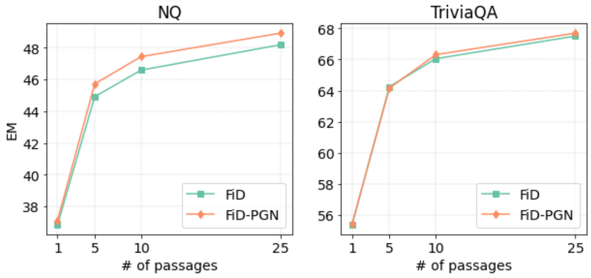


Figure 3: The variation of performance with different number of retrieved passages used in reader training. Exact match (EM) scores are measured on the development sets of NQ and TriviaQA.

sages. We can observe that the matching scores of both models increase with respect to the number of passages used in training, consistent with the findings in Izacard and Grave (2021b) that sequence-to-sequence model is capable of gathering information across multiple retrieved passages. Moreover, the two models show comparative performance when the number of training passages is small, but when more passages are included, our model outperforms FiD, especially on the NQ dataset.

6 Conclusion

In this article, we propose a novel FiD-PGN approach for the reader module of ODQA under the standard *retriever-reader* framework. Specifically, we integrate a pointer network into the FiD reader to allow the model to directly select words from the retrieved passages. Experimental results show that our model outperforms FiD-KD on two benchmark datasets under the same setting, demonstrating the advantages of our method.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

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

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

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Gautier Izacard and Edouard Grave. 2021a. Distilling knowledge from reader to retriever for question answering. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Gautier Izacard and Edouard Grave. 2021b. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

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

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

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1000–1008, Online. Association for Computational Linguistics.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv preprint*, abs/1711.05101.

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 11–19, Beijing, China. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Sewon Min, Jordan L. Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, Colin Raffel, Adam Roberts, Tom Kwiatkowski, Patrick S. H. Lewis, Yuxiang Wu, Heinrich Küttler, Linqing Liu, Pasquale Minervini, Pontus Stenetorp, Sebastian Riedel, Sohee Yang, Minjoon Seo, Gautier Izacard, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Edouard Grave, Ikuya Yamada, Sonse Shimaoka, Masatoshi Suzuki, Shumpei Miyawaki, Shun Sato, Ryo Takahashi, Jun Suzuki, Martin Fajcik, Martin Docekal, Karel Ondrej, Pavel Smrz, Hao Cheng, Yelong Shen, Xiaodong Liu, Pengcheng He, Weizhu Chen, Jianfeng Gao, Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Sejr Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Wen-tau Yih. 2021a. Neurips 2020 efficientqa competition: Systems, analyses and lessons learned. *ArXiv preprint*, abs/2101.00133.

Sewon Min, Kenton Lee, Ming-Wei Chang, Kristina Toutanova, and Hannaneh Hajishirzi. 2021b. Joint passage ranking for diverse multi-answer retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6997–7008, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv preprint*, abs/1910.10683.

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

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2021. QA dataset explosion: A taxonomy of NLP resources for question answering and reading comprehension. *ArXiv preprint*, abs/2107.12708.

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

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural*

Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada, pages 2692–2700.

Ellen M. Voorhees. 1999. The TREC-8 question answering track report. In *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999*, volume 500-246 of *NIST Special Publication*. National Institute of Standards and Technology (NIST).

Yuxiang Wu, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2021. Training adaptive computation for open-domain question answering with computational constraints. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 447–453, Online. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, and Hannaneh Hajishirzi. 2021. Efficient passage retrieval with hashing for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 979–986, Online. Association for Computational Linguistics.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota. Association for Computational Linguistics.

Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Francisco Guzmán, Luke Zettlemoyer, and Marjan Ghazvininejad. 2021. Detecting hallucinated content in conditional neural sequence generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1393–1404, Online. Association for Computational Linguistics.

Augmenting Document Representations for Dense Retrieval with Interpolation and Perturbation

Soyeong Jeong[1] **Jinheon Baek**[2] **Sukmin Cho**[1] **Sung Ju Hwang**[2] **Jong C. Park**[1*]
School of Computing[1] Graduate School of AI[2]
Korea Advanced Institute of Science and Technology[1,2]
`{syjeong,nelllpic,park}@nlp.kaist.ac.kr`[1]
`{jinheon.baek,sjhwang82}@kaist.ac.kr`[2]

Abstract

Dense retrieval models, which aim at retrieving the most relevant document for an input query on a dense representation space, have gained considerable attention for their remarkable success. Yet, dense models require a vast amount of labeled training data for notable performance, whereas it is often challenging to acquire query-document pairs annotated by humans. To tackle this problem, we propose a simple but effective **D**ocument **A**ugmentation for dense **R**etrieval (DAR) framework, which augments the representations of documents with their interpolation and perturbation. We validate the performance of DAR on retrieval tasks with two benchmark datasets, showing that the proposed DAR significantly outperforms relevant baselines on the dense retrieval of both the labeled and unlabeled documents.

1 Introduction

Retrieval systems aim at retrieving the documents most relevant to the input queries, and have received substantial spotlight since they work as core elements in diverse applications, especially for open-domain question answering (QA) (Voorhees, 1999). Open-domain QA is a task of answering the question from a massive amount of documents, often requiring two components, a retriever and a reader (Chen et al., 2017; Karpukhin et al., 2020). Specifically, a retriever ranks the most question-related documents, and a reader answers the question using the retrieved documents.

Traditional sparse retrieval approaches such as BM25 (Robertson et al., 1994) and TF-IDF rely on term-based matching, hence suffering from the vocabulary mismatch problem: the failure of retrieving relevant documents due to the lexical difference from queries. To tackle such a problem, recent research focuses on dense retrieval models to generate learnable dense representations for queries and documents (Karpukhin et al., 2020).

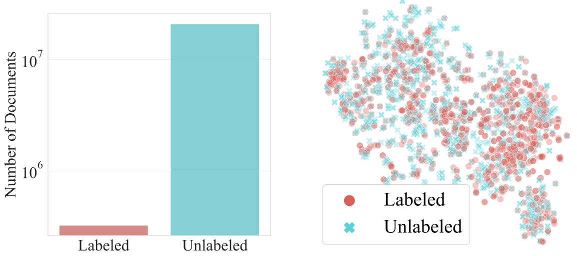


Figure 1: (Left) The number of labeled and unlabeled documents for the Natural Question dataset. (Right) T-SNE (Maaten and Hinton, 2008) visualization of randomly sampled document representations from the DPR model.

Despite their recent successes, some challenges still remain in the dense retrieval scheme for a couple of reasons. First, dense retrieval models need a large amount of labeled training data for a decent performance. However, as Figure 1 shows, the proportion of labeled query-document pairs is extremely small since it is almost impossible to rely on humans for the annotations of a large document corpus. Second, in order to adapt a retrieval model to the real world, where new documents constantly emerge, handling unlabeled documents that are not seen during training should obviously be considered, but remains challenging.

To automatically expand the query-document pairs, recent work generates queries from generative models (Liang et al., 2020; Ma et al., 2021) or incorporates queries from other datasets (Qu et al., 2021), and then generates extra pairs of augmented queries and documents. However, these query augmentation schemes have serious and obvious drawbacks. First, it is infeasible to augment queries for every document in the dataset (see the number of unlabeled documents in Figure 1), since generating and pairing queries are quite costly. Second, even after obtaining new pairs, we need extra training steps to reflect the generated pairs on the retrieval model. Third, this query augmentation method does not add variations to the documents but only to the queries, thus it may be suboptimal to handle enormous unlabeled documents.

[*] Corresponding author

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 442 - 452
May 22-27, 2022 ©2022 Association for Computational Linguistics

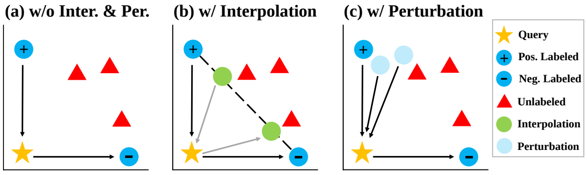


Figure 2: Our document augmenting schemes of interpolation and perturbation on a dense representation space. Pos. and Neg. denote positive and negative documents to the query.

Since augmenting additional queries is costly, the question is then if it is feasible to only manipulate the given query-document pairing to handle numerous unlabeled documents. To answer this, we first visualize the embeddings of labeled and unlabeled documents. Figure 1 shows that there is no distinct distributional shift between labeled and unlabeled documents. Thus it could be effective to manipulate only the labeled documents to handle the nearby unlabeled documents as well as the labeled documents. Using this observation, we propose a novel document augmentation method for a dense retriever, which not only interpolates two different document representations associated with the labeled query (Figure 2 (b)), but also stochastically perturbs the representations of labeled documents with a dropout mask (Figure 2 (c)). One notable advantage of our scheme is that, since it manipulates only the representations of documents, our model does not require explicit annotation steps of query-document pairs, which makes it highly efficient. We refer to our overall method as Document Augmentation for dense Retrieval (DAR).

We experimentally validate our method on standard open-domain QA datasets, namely Natural Question (NQ) (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017) (TQA), against various evaluation metrics for retrieval models. The experimental results show that our method significantly improves the retrieval performances on both the unlabeled and labeled documents. Furthermore, a detailed analysis of the proposed model shows that interpolation and stochastic perturbation positively contribute to the overall performance.

Our contributions in this work are threefold:

- We propose to augment documents for dense retrieval models to tackle the problem of insufficient labels of query-document pairs.
- We present two novel document augmentation schemes for dense retrievers: interpolation and perturbation of document representations.
- We show that our method achieves outstanding retrieval performances on both labeled and unlabeled documents on open-domain QA tasks.

2 Related Work

Dense Retriever Dense retrieval models (Lee et al., 2019; Karpukhin et al., 2020) have gained much attention, which generate dense representations for queries and documents. However, dense retrieval faces a critical challenge from limited training data. Recent work has addressed such a problem by generating extra query-document pairs to augment those pairs to the original dense retrieval model (Liang et al., 2020; Ma et al., 2021; Qu et al., 2021), or by regularizing the model (Rosset et al., 2019). However, unlike ours that automatically augments data during a training phase, these methods require extensive computational resources for an additional generation step of explicitly query-document pairing before training the retriever.

Data Augmentation Since data augmentation is crucial to the performance of deep neural networks, it is widely applied to diverse domains (Shorten and Khoshgoftaar, 2019; Hedderich et al., 2021), where interpolation and perturbation are dominant methods. Mixup interpolates two items, such as pixels of images, to augment the training data (Zhang et al., 2018; Verma et al., 2019), which is also adopted for NLP (Chen et al., 2020; Yin et al., 2021). However, none of the previous work has shown the effectiveness of mixup when applied to retrieval tasks. Besides interpolation, Wei and Zou (2019) and Ma (2019) proposed perturbation over words, and Lee et al. (2021b) proposed perturbation over word embeddings. Jeong et al. (2021) and Gao et al. (2021) perturbed text embeddings to generate diverse sentences and to augment positive sentence pairs in unsupervised learning. In contrast, we address dense retrieval, perturbing document representations with dropout (Srivastava et al., 2014) in a supervised setting with labeled documents.

3 Method

We begin with the definition of dense retrieval.

Dense Retrieval Given a pair of query q and document d, the goal of dense retrieval is to correctly calculate a similarity score between them from the dense representations $\boldsymbol{q}$ and $\boldsymbol{d}$, as follows:

$$\begin{aligned} & f(q,d) = \text{sim}(\boldsymbol{q}, \boldsymbol{d}), \\ & \boldsymbol{q} = E_Q(q;\theta_q) \quad \text{and} \quad \boldsymbol{d} = E_D(d;\theta_d), \end{aligned} \tag{1}$$

where f is a scoring function that measures the similarity between a query-document pair, sim is a

	Natural Questions (NQ)						TriviaQA (TQA)					
	MRR	MAP	T-100	T-20	T-5	T-1	MRR	MAP	T-100	T-20	T-5	T-1
BM25	32.46	20.78	78.25	62.94	43.77	22.11	55.28	34.85	83.15	76.41	66.28	46.30
DPR	39.55	25.61	83.77	72.94	54.02	27.45	44.29	27.24	80.50	71.07	57.74	33.63
DPR w/ QA	40.00	24.93	83.46	72.13	55.46	27.67	46.27	28.08	80.76	71.88	59.14	35.90
DPR w/ DA	41.28	26.60	83.68	72.83	55.51	29.31	46.08	27.82	80.42	71.55	58.64	35.85
DPR w/ AR	41.18	26.04	83.60	73.41	55.51	29.11	45.13	27.57	80.65	71.68	58.09	34.52
DAR (Ours)	**42.92**	**27.12**	**84.18**	**75.04**	**57.62**	**30.42**	**47.32**	**28.70**	**81.30**	**72.66**	**59.88**	**36.94**
QAR (Ours)	**43.09**	**27.64**	**84.21**	**74.76**	**57.51**	**31.25**	**47.21**	**29.00**	**80.91**	**72.12**	**59.94**	**36.92**

Table 1: Retrieval results on NQ and TQA datasets, including the variant of our model – QAR: applying data augmentation techniques to queries instead of documents. BM25 is the sparse retrieval model, whereas others are dense retrieval models. The best model and the second best model among dense retrievers are denoted in **bold**, which we aim to improve in this work.

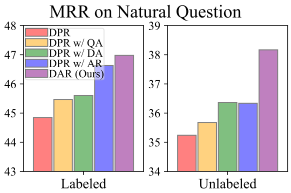


Figure 3: Retrieval results on the labeled and unlabeled documents in the NQ dataset with MRR as an evaluation metric.

similarity metric such as cosine similarity, and E_Q and E_D are dense encoders for a query and document, respectively, with parameters $\theta = (\theta_q, \theta_d)$.

A dense retrieval scheme generally uses the negative sampling strategy to distinguish the relevant query-document pairs from irrelevant pairs, which generates an effective representation space for queries and documents. We specify a relevant query-document pair as $(q, d^+) \in \tau^+$, and an irrelevant pair as $(q, d^-) \in \tau^-$, where $\tau^+ \cap \tau^- = \emptyset$. The objective function is as follows:

$$\min_{\theta} \sum_{(q,d^+)\in\tau^+} \sum_{(q,d^-)\in\tau^-} \mathcal{L}(f(q, d^+), f(q, d^-)), \quad (2)$$

where a loss function $\mathcal{L}$ is a negative log-likelihood of the positive document. Our goal is to augment a set of query-document pairs, by manipulating documents with their interpolation or perturbation, which we explain in the next paragraphs.

Interpolation with Mixup As shown in interpolation of Figure 2, we aim at augmenting the document representation located between two labeled documents to obtain more query-document pairs, which could be useful to handle unlabeled documents in the middle of two labeled documents. To achieve this goal, we propose to interpolate the positive and negative documents (d^+, d^-) for the given query q, adopting mixup (Zhang et al., 2018). Note that, since the input documents to the encoder E_D are discrete, we use the output embeddings of documents to interpolate them, as follows:

$$\tilde{\boldsymbol{d}} = \lambda \boldsymbol{d}^+ + (1 - \lambda)\boldsymbol{d}^-, \quad (3)$$

where $\tilde{\boldsymbol{d}}$ is the mixed representation of positive and negative documents for the given query q, and $\lambda \in [0, 1]$. We then optimize the model to estimate the similarity sim$(\boldsymbol{q}, \tilde{\boldsymbol{d}})$ between the interpolated document and the query as the soft label λ with a binary cross-entropy loss. The output of the cross-entropy loss is added to the original loss in equation 2. One notable advantage of our scheme is that the negative log-likelihood loss in equation 2 maximizes the similarity score of the positive pair, while minimizing the score of the negative pair; thus there are no intermediate similarities between arbitrary query-document pairs. However, ours can obtain query-document pairs having soft labels, rather than strict positive or negative classes, by interpolating the positive and negative documents.

# Query		MRR	R@1k
10K	ANCE	42.62	94.60
	+ DAR	**46.31**	**94.81**
50K	ANCE	46.88	**95.58**
	+ DAR	**48.20**	**95.58**

Table 2: Results on the MS MARCO subsets with ANCE as a denser retriever.

	Time (Min.)	Memory (MiB)
DPR	19	22,071
DPR w/ QA	41	22,071
DPR w/ DA	38	22,071
DPR w/ AR	29	38,986
DAR (Ours)	21	22,071

Table 3: Wall-clock time and maximum memory usage for training a DPR model per epoch.

Stochastic Perturbation with Dropout In addition to our interpolation scheme to handle unlabeled documents in the space of interpolation of two labeled documents, we further aim at perturbing the labeled document to handle its nearby unlabeled documents as shown in Figure 2 (c). In order to do so, we randomly mask the representation of the labeled document, obtained by the document encoder E_D, with dropout, where we sample masks from a Bernoulli distribution. In other words, if we sample n different masks from the distribution, we obtain n different query-document pairs $\left\{(\boldsymbol{q}, \boldsymbol{d}_i^+)\right\}_{i=1}^{i=n}$ from one positive pair $(\boldsymbol{q}, \boldsymbol{d}^+)$. By doing so, we augment n times more positive query-document pairs by replacing a single positive pair (q, d^+) in equation 2. Moreover, since the document perturbation is orthogonal to the interpolation, we further interpolate between the perturbed positive document $\boldsymbol{d}_i^+$ and the negative document $\boldsymbol{d}^-$ for the given query in equation 3, to augment a soft query-document pair from perturbation.

Efficiency Data augmentation methods are generally vulnerable to inefficiency, since they need a vast amount of resources to generate data and to forward the generated data into the large language model. However, since our interpolation and perturbation methods only manipulate the already

	MRR	MAP	T-20	T-5
DAR (Ours)	**42.92**	**27.12**	**75.04**	**57.62**
w/o Perturbation	41.26	26.19	73.68	55.37
w/o Interpolation	40.40	25.70	73.41	55.29
DPR	39.55	25.61	72.94	54.02

Table 4: Ablation studies of our DAR on the NQ dataset by removing interpolation or perturbation.

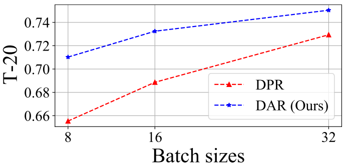


Figure 4: T-20 on the NQ dataset with varying batch sizes.

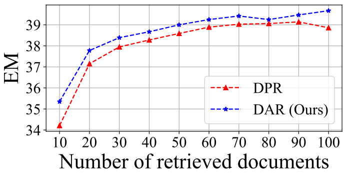


Figure 5: Exact Match (EM) scores for a reader on the NQ.

obtained representations of the documents from the encoder E_D, we don't have to newly generate document texts and also to forward generated documents into the model, which greatly saves time and memory (see Table 3). We provide a detailed analysis and discussion of efficiency in **Appendix** B.1.

4 Experiments

4.1 Experimental Setups

Here, we describe datasets, models, and implementation details for experiments. More experimental details are shown in **Appendix** A. Our code is publicly available at github.com/starsuzi/DAR.

Datasets For documents to retrieve, we use the Wikipedia, following Karpukhin et al. (2020), where the processed dataset contains 21,015,324 passages. To evaluate retrieval models, we use two open-domain QA datasets, following Karpukhin et al. (2020): 1) **Natural Questions (NQ)** is collected with Google search queries (Kwiatkowski et al., 2019); 2) **TriviaQA (TQA)** is a QA collection scraped from the Web (Joshi et al., 2017).

Retrieval Models **1) BM25** is a sparse term-based retrieval model based on TF-IDF (Robertson et al., 1994). **2) Dense Passage Retriever (DPR)** is a dense retrieval model with a dual-encoder of query-document pairs (Karpukhin et al., 2020). **3) DPR with Query Augmentation (DPR w/ QA)** augments pairs with query generation for the document, adopting (Liang et al., 2020; Mao et al., 2021a). **4) DPR with Document Augmentation (DPR w/ DA)** augments pairs by replacing words in the document (Ma, 2019). **5) DPR with Axiomatic Regularization (DPR w/ AR)** regularizes the retrieval model to satisfy certain axioms (Rosset et al., 2019). **6) DAR** is ours with interpolation and perturbation of document representations.

Metrics **1) Top-K Accuracy (T-K)** computes whether a query's answer is included in Top-K retrieved documents. **2) Mean Reciprocal Rank (MRR)** and **3) Mean Average Precision (MAP)** measure the first rank and the average precision of query-relevant retrieved documents, respectively.

Implementation Details For the dense retrieval model based on the DPR framework, we refer to the publicly available code from DPR (Karpukhin et al., 2020). We set the training epoch as 25 and batch size as 32 under academic budgets with a single GeForce RTX 3090 GPU having 24GB memory. We use in-batch negative sampling as our negative sampling strategy without hard negative samples. Also, we retrieve 100 passages per question.

We use both interpolation and perturbation schemes for our augmentation methods. Specifically, for the interpolation method, we set $\lambda \in [0,1]$ in equation 3 to be sampled from the uniform distribution. Also, for the perturbation method, we set the dropping rate as 0.1, and the number of dropout masks n is selected in the range of 3 to 9.

4.2 Results

In this subsection, we show the overall performance of our DAR, and then give detailed analyses.

Overall Results As Table 1 shows, DAR outperforms dense retrieval baselines on all datasets on the DPR framework. Note that DAR contributes to more accurate retrieval performance, since the smaller K gives higher performance improvements. Furthermore, Figure 3 shows that, with our method, the retrieval performance on unlabeled documents – not seen during training – together with the labeled ones is improved, where performance gains on unlabeled are remarkable. To see the robustness of DAR on other retrievers, we further evaluate our model on the recent ANCE framework (see **Appendix** A for setups). As Table 2 shows, we observe that the performance improvement is more dominant on MRR when given a smaller number of training queries (low-resource settings), that DAR effectively augments document representations.

Results on Query Augmentation We focus on the problem of a notably small proportion of labeled documents in the training dataset, and propose to augment representations of unlabeled documents, which are not seen during training. However, it is also possible to augment representations of queries – likely to be unseen at the test time

	MRR	MAP	T-100	T-1
DPR+HN	53.40	33.38	84.82	43.21
DAR+HN (Ours)	**54.18**	**33.71**	**85.35**	**44.18**

Table 5: Retrieval results with hard negatives (HN) from BM25 on the NQ dataset for the DPR framework.

– by applying our interpolation and perturbation methods directly to queries. Note that we refer to our query augmentation method as Query Augmentation for dense Retrieval (QAR). As shown in Table 1, our proposed augmentation strategies also effectively improve the retrieval performance even when applied to queries. This result implies that our method is versatile, regardless of whether it is applied to documents or queries.

Effectiveness of Interpolation & Perturbation To understand how much our proposed interpolation and perturbation techniques contribute to the performance gain, we perform ablation studies. Table 4 shows that each of the interpolation and stochastic perturbation positively contributes to the performance. In particular, when both of them are simultaneously applied, the performance is much improved, which demonstrates that these two techniques are in a complementary relationship.

Batch Size We test DAR with varying numbers of batch sizes. Figure 4 indicates that our DAR consistently improves the retrieval performance. Note that the smaller the batch size, the bigger the performance gap. Also, the batch size 16 of DAR outperforms the batch size 32 of the baseline, which highlights that DAR effectively augments document representations with a small batch.

Reader Performance To see whether accurately retrieved documents lead to better QA performance, we experiment with the same extractive reader from DPR without additional re-training. Figure 5 illustrates the effectiveness of our method on passage reading with varying numbers of retrieved documents. We observe that our retrieval result with small retrieved documents (i.e., $K = 10$) significantly improves the performance of the reader. This implies that a more accurate retrieval on smaller K in Table 1 helps achieve the improved QA performance as Lee et al. (2021a) described. Furthermore, our reader performance may be further enhanced with advanced reading schemes (Mao et al., 2021a; Qu et al., 2021; Mao et al., 2021b).

Negative Sampling Strategy To see the effectiveness of our DAR coupled with an advanced negative sampling scheme, we compare DAR against the baseline with the hard negative sampling strategy from BM25 (Karpukhin et al., 2020). Table 5 shows that DAR with hard negative sampling outperforms the baseline method. The results demonstrate that the performance of dense retrieval models could be further strengthened with a combination of our augmentation methods and advanced negative sampling techniques. Also, in all our experiments of the ANCE framework, we already use the strategy of negative sampling in Xiong et al. (2021), where we observe the clear performance improvement of our DAR on ANCE in Table 2.

	MRR	MAP	T-100	T-20	T-5	T-1
BM25	29.60	28.05	77.87	61.30	42.27	18.86
DPR	31.79	29.94	88.30	70.48	45.48	19.18
DPR w/ QA	30.02	28.26	86.82	68.80	43.95	17.56
DPR w/ DA	31.96	30.25	87.75	71.29	46.55	19.03
DPR w/ AR	31.41	29.50	88.27	70.57	45.10	19.12
DAR (Ours)	**33.37**	**31.49**	**88.93**	**73.70**	**48.38**	**20.16**

Table 6: Retrieval results on the NQ dataset, following the processing procedure of Thakur et al. (2021).

Results on Different Data Processing We additionally evaluate DAR on another NQ test dataset, following the processing procedure of Thakur et al. (2021). For experiments, we reuse the same training checkpoint used in Table 1, as the training dataset is equal across the settings of Karpukhin et al. (2020) and Thakur et al. (2021). As Table 6 shows, our DAR also consistently outperforms all baselines when tested on the NQ test set from Thakur et al. (2021). This confirms that our DAR robustly improves retrieval performances, regardless of the specific data processing strategies.

5 Conclusion

We presented a novel method of augmenting document representations focusing on dense retrievers, which require an extensive amount of labeled query-document pairs for training. Specifically, we augment documents by interpolating and perturbing their embeddings with mixup and dropout masks. The experimental results and analyses on multiple benchmark datasets demonstrate that DAR greatly improves retrieval performances.

Acknowledgements

This work was supported by Institute for Information and communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00582, Prediction and augmentation of the credibility distribution via linguistic analysis and automated evidence document collection).

Ethical Statements

Retrieving the most relevant documents from the user's query is increasingly important in a real-world setting, as it is widely used from web search, to question answering, to dialogue generation systems. Notably, our work contributes to the accurate retrieval of documents with the proposed data augmentation strategies, thus improving the document retrieval performances on real-world applications. However, we have to still consider the failure of retrieval systems on low-resource but high-risk domains (e.g., biomedicine), where the labeled data for training retrieval models is limited yet one failure can yield a huge negative impact. While we strongly believe that our data augmentation strategies – interpolation and perturbation of document representations – are also helpful to improve the retrieval performances on such low-resource domains, the model's prediction performance is still far from perfect, and more efforts should be made to develop a reliable system.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1533–1544. ACL.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1870–1879. Association for Computational Linguistics.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. MixText: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2147–2157, Online. Association for Computational Linguistics.

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

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Soyeong Jeong, Jinheon Baek, ChaeHun Park, and Jong Park. 2021. Unsupervised document expansion for information retrieval with stochastic text generation. In *Proceedings of the Second Workshop on Scholarly Document Processing*, pages 7–17, Online. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

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

Jinhyuk Lee, Alexander Wettig, and Danqi Chen. 2021a. Phrase retrieval learns passage retrieval, too. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Seanie Lee, Minki Kang, Juho Lee, and Sung Ju Hwang. 2021b. Learning to perturb word embeddings for out-of-distribution qa. In *Proceedings of* *the 58th Annual Meeting of the Association for Computational Linguistics*.

Davis Liang, Peng Xu, Siamak Shakeri, Cicero Nogueira dos Santos, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Embedding-based zero-shot retrieval through query generation. *arXiv preprint arXiv:2009.10270*.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald. 2021. Zero-shot neural passage retrieval via domain-targeted synthetic question generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1075–1088, Online. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021a. Generation-augmented retrieval for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4089–4100, Online. Association for Computational Linguistics.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021b. Reader-guided passage reranking for open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 344–350, Online. Association for Computational Linguistics.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5835–5847.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference, TREC 1994, Gaithersburg, Maryland, USA, November 2-4, 1994*, volume 500-225 of *NIST Special Publication*, pages 109–126. National Institute of Standards and Technology (NIST).

Corby Rosset, Bhaskar Mitra, Chenyan Xiong, Nick Craswell, Xia Song, and Saurabh Tiwary. 2019. An axiomatic approach to regularizing neural ranking models. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 981–984. ACM.

Connor Shorten and Taghi M. Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *J. Big Data*, 6:60.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019. Manifold mixup: Better representations by interpolating hidden states. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6438–6447. PMLR.

Ellen M. Voorhees. 1999. The TREC-8 question answering track report. In *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999*, volume 500-246 of *NIST Special Publication*. National Institute of Standards and Technology (NIST).

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389, Hong Kong, China. Association for Computational Linguistics.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

Wenpeng Yin, Huan Wang, Jin Qu, and Caiming Xiong. 2021. BatchMixup: Improving training by interpolating hidden states of the entire mini-batch. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4908–4912, Online. Association for Computational Linguistics.

Yang You, Jing Li, Sashank J. Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh.

2020. Large batch optimization for deep learning: Training BERT in 76 minutes. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. *International Conference on Learning Representations*.

	Train	Val	Test
Natural Question (NQ)	58,880	6,515	3,610
TriviaQA (TQA)	60,413	6,760	11,313
MS MARCO, # Query: 10K	6,591	6,980	-
MS MARCO, # Query: 50K	32,927	6,980	-

Table 7: Statistics for training, validation, and test sets on the NQ, TQA, and randomly sampled MS MARCO datasets. Note that, for MS MARCO, we only sample the number of training query-document pairs except for the validation set.

A Experimental Setups

Datasets To evaluate the performance of retrieval models, we need two types of datasets: 1) a set of documents to retrieve, and 2) pairs of a query and a relevant document, having an answer for the query. We first explain the datasets that we used for the DPR framework (Karpukhin et al., 2020), and then describe the dataset for the ANCE framework (Xiong et al., 2021).

For documents to retrieve, we use the Wikipedia snapshot from December 20, 2018, which contains 21,015,324 passages consisting of 100 tokens, following Karpukhin et al. (2020) for the DPR framework. For open-domain QA datasets, we use Natural Question (NQ) (Kwiatkowski et al., 2019) and Trivia QA (TQA) (Joshi et al., 2017), following the dataset processing procedure of Karpukhin et al. (2020). We report the statistics of the training, validation, and test sets on NQ and TQA in Table 7.

To see the performance gain of our DAR on other dense retrieval models, we evaluate DAR on the ANCE framework (Xiong et al., 2021), which is one of the recent dense retrieval models. ANCE is evaluated on the MS MARCO dataset, thus we use MS MARCO for training and testing our model. Note that training ANCE with the full MS MARCO dataset requires 225 GPU hours even after excluding the excessive BM25 pre-training and inference steps. Thus we randomly sample the MS MARCO dataset to train the model under academic budgets. Specifically, the subset of our MS MARCO passage dataset contains 500,000 passages. Also, we randomly divide the training queries into two subsets: one for 10,000 training queries and the other for 50,000 training queries. Then we align the sampled training queries to the query-document pairs in the MS MARCO dataset. On the other hand, we do not modify the validation set (dev set) of query-document pairs for testing. We summarize the statistics of the dataset in Table 7. Note that since the test set of MS MARCO is not publicly open, we evaluate the dense retrievers with the validation set, following Xiong et al. (2021).

Metrics Here, we explain the evaluation metrics for retrievers in detail. Specifically, given an input query, we measure the ranks of the correctly retrieved documents for the DPR framework with the following metrics:

1) Top-K Accuracy (T-K): It measures whether an answer of the given query is included in the retrieved Top-K documents.

2) Mean Reciprocal Rank (MRR): It computes the rank of the first correct document for the given query among the Top-100 retrieved documents, and then computes the average of the reciprocal ranks for all queries.

3) Mean Average Precision (MAP): It computes the mean of the average precision scores for all queries, where precision scores are calculated by the ranks of the correctly retrieved documents among Top-100 ranked documents.

We use the following evaluation metric for the reader, which identifies the answer from retrieved documents.

1) Exact Match (EM): It measures whether the reader exactly predicts one of the reference answers for each question.

Note that, for the ANCE framework, we follow the evaluation metrics, namely MRR@10 and Recall@1k, in the original paper (Xiong et al., 2021).

Experimental Implementation Details For dense retrieval models based on the DPR framework, we follow the dual-encoder structure of query and document by using the publicly available code from DPR[1] (Karpukhin et al., 2020). For all experiments, we set the batch size as 32, and train models on a single GeForce RTX 3090 GPU having 24GB memory. Note that, in contrast to the best reported setting of DPR which requires industrial-level resources of 8 V100 GPUs (8 × 32GB = 256GB) for training with a batch size of 128, we use a batch size of 32 to train the model under academic budgets. We optimize the model parameters of all dense retrieval models with the Adam optimizer (Kingma and Ba, 2015) having a learning rate of 2e-05. We train the models for 25 epochs, following the analysis[2] that the training phases converge after 25 epochs.

For the retrievers based on the ANCE framework, we refer to the implementation from ANCE[3] (Xiong et al., 2021). In order to directly

[1] https://github.com/facebookresearch/DPR

[2] See footnote 1.

[3] https://github.com/microsoft/ANCE

measure the performance gain of the dense retrieval models based on ANCE from using our DAR, we use the pre-trained RoBERTa without warming up with the BM25 negatives. We train all the dense retrieval models for 50,000 steps with a single GeForce RTX 3090 GPU having 24GB memory, and simultaneously generate the ANN index with another GeForce RTX 3090 GPU, following Xiong et al. (2021). Following the standard implementation setting, we set the training batch size as 8, and optimize the model with the LAMB optimizer (You et al., 2020) with a learning rate of 1e-6.

Architectural Implementation Details For our augmentation methods, we use both interpolation and perturbation schemes of document representations obtained from the document encoder E_D in equation 1. Specifically, given a positive query-document pair $(\boldsymbol{q}, \boldsymbol{d}^+)$, we first perturb the document representation $\boldsymbol{d}^+$ with dropout masks sampled from a Bernoulli distribution, which generates n numbers of perturbed document representations $\{\boldsymbol{d}_i^+\}_{i=1}^{i=n}$. Then, we augment them to generate n numbers of positive query-document pairs $\{(\boldsymbol{q}, \boldsymbol{d}_i^+)\}_{i=1}^{i=n}$, which we use in equation 2. We search the number of perturbations n in the range from 3 to 9, and set the probability of the Bernoulli distribution as 0.1.

Instead of only using positive or negative pairs, we further augment query-document pairs having intermediate similarities with mixup. Specifically, we interpolate representations between the perturbed-positive document $\boldsymbol{d}_i^+$ and the negative document $\boldsymbol{d}^-$ for the given query $\boldsymbol{q}$, with $\lambda \in [0, 1]$ in equation 3 sampled from a uniform distribution. Note that, given a positive pair of a query and a document, we consider the documents not identified as positive in the batch as negative documents. In other words, if we set the batch size as 32, then we could generate 31 interpolated document representations from 1 positive pair and 31 negative pairs. To jointly train the interpolation scheme with the original objective, we add the loss obtained from interpolation to the loss in equation 2.

B Additional Experimental Results

B.1 Efficiency

As described in the Efficiency paragraph of Section 3, compared to the existing query augmentation methods (Liang et al., 2020; Ma et al., 2021; Qu et al., 2021), document augmentation method (Ma, 2019), and word replacement method for regularization (Rosset et al., 2019), our method of augmenting document representations with interpolation and perturbation in a dense representation space is highly efficient. This is because, unlike the baselines above, we do not explicitly generate or replace a query or document text; but rather we only manipulate the representations of documents. This scheme greatly saves the time for training, since additional forwarding of the generated or replaced query-document pairs into the language model is not required for our data augmentation methods.

To empirically validate the efficiency of our methods against the baselines, we report the memory usage and time for training a retrieval model per epoch in Table 3. As for memory efficiency, all the compared dense retrieval models using data augmentation methods, including ours, use the same amount of maximum GPU memory. This shows that the overhead of memory usage comes from operations in the large-size language model, such as BERT (Devlin et al., 2019), not from manipulating the obtained document representations to augment the query-document pairs. Technically speaking, there are no additional parameters to augment document representations; thus our interpolation and perturbation methods do not increase the memory usage. On the other hand, DPR w/ AR excessively increases the memory usage, since it requires an extra forwarding process to the language model to represent the additional word-replaced sentences for regularization, instead of using the already obtained dense representations like ours.

We also report the training time for dense retrievers in Table 3. Note that, for the explicit augmentation method based models, such as DPR w/ QA and DPR w/ DA, we exclude the extra time for training a generation model and generating a query or document for the given text. Also, we additionally generate the same number of query-document pairs in the training set, where the total amount of training data-points for DPR w/ QA and DPR w/ DA baselines are twice larger than the original dataset. Unlike these explicit query or document generation baselines, we perturb the document n times, but also interpolate the representations of positive and negative documents. As shown in Table 3, our DAR is about doubly more efficient than the explicit text augmentation methods, since DPR w/ QA and DPR w/ DA explicitly augment query-document pairs instead of using the obtained dense

	T-5	T-20	T-100
DPR (Karpukhin et al., 2020)	52.1	70.8	82.1
DPR (Ours)	**53.2**	**71.6**	**82.7**

Table 8: Comparison of the DPR models' Top-K accuracy between the reported and reproduced scores. Best performance is highlighted in **bold**.

representations like ours. Also, our DAR takes a little more time to augment document representations than the base DPR model, while significantly improving retrieval performances as shown in Table 1. Even compared to the term replacement based regularization model (DPR w/ AR), our DAR shows noticeable efficiency, since an additional embedding process of the document after the word replacement on it requires another forwarding step besides the original forwarding step.

B.2 Reproduction of DPR

We strictly set the batch size as 32 for training all the dense retrievers using the DPR framework; therefore the retrieval performances are different from the originally reported ones in Karpukhin et al. (2020) that use a batch size of 128. However, while we use the available code from the DPR paper, one may wonder if our reproduction result is accurate. Therefore, since Karpukhin et al. (2020) provided the retrieval performances of the DPR with different batch sizes (e.g., a batch size of 32), evaluated on the development (validation) set of the NQ dataset, we compare the Top-K accuracy between the reported scores and our reproduced scores. Table 8 shows that our reproduced Top-K accuracy scores with three different Ks (e.g., Top-5, Top-20, and Top-100) are indeed similar to the reported ones, with ours even higher, thus showing that our reproductions are accurate.

B.3 Experiment on WebQuestions

One may have a concern that, as a sparse retrieval model – BM25 – outperforms all the other dense retrieval models on the TQA dataset in Table 1, TQA is not good enough to demonstrate the strength of our dense augmentation strategy. While we believe that sparse retrieval models are not our competitors as we aim to improve the dense retrieval models with data augmentation, in order to clear out such a concern, we additionally train and evaluate our DAR on the WebQuestions (WQ) dataset (Berant et al., 2013), following the data processing procedure from (Karpukhin et al., 2020). As Table 9 shows, our DAR outperforms both dense and sparse retrieval models. Thus, the best scheme among sparse and dense retrievers still depends on the dataset, and combining sparse and dense models to complement each other will be a valuable research direction, which we leave as future work.

	MRR	MAP	T-100	T-20	T-5	T-1
BM25	29.75	19.15	75.49	62.40	41.83	18.90
DPR	33.34	21.76	78.64	65.75	45.87	22.00
DAR (Ours)	**34.48**	**22.16**	**78.79**	**67.37**	**47.54**	**23.23**

Table 9: Retrieval results on the WQ dataset, in which the best performance is highlighted in **bold**.

WLASL-LEX: a Dataset for Recognising Phonological Properties in American Sign Language

Federico Tavella and **Viktor Schlegel** and **Marta Romeo**
Aphrodite Galata and **Angelo Cangelosi**
{name.surname}@manchester.ac.uk
Department of Computer Science, The University of Manchester

Abstract

Signed Language Processing (SLP) concerns the automated processing of signed languages, the main means of communication of Deaf and hearing impaired individuals. SLP features many different tasks, ranging from sign recognition to translation and production of signed speech, but it has been overlooked by the NLP community thus far. In this paper, we bring attention to the task of modelling the phonology of sign languages. We leverage existing resources to construct a large-scale dataset of American Sign Language signs annotated with six different phonological properties. We then conduct an extensive empirical study to investigate whether data-driven end-to-end and feature-based approaches can be optimised to automatically recognise these properties. We find that, despite the inherent challenges of the task, graph-based neural networks that operate over skeleton features extracted from raw videos are able to succeed at the task to a varying degree. Most importantly, we show that this performance pertains even on signs unobserved during training.

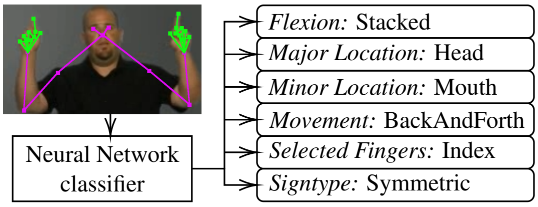


Figure 1: We annotate ASL sign videos with their corresponding phonological information and skeleton features of the speakers, and train neural networks to recognise the former from the latter.

1 Introduction

Around 200 languages in the world are signed rather than spoken, featuring their own vocabulary and grammatical structures. For example the American Sign Language (ASL) is not a mere translation of English into signs and is unrelated to the British Sign Language (BSL). Their non-textual nature introduces many challenges to their automated processing, compared with purely textual NLP. Research on Sign Language Processing (SLP) encompasses tasks such as sign language detection, i.e. recognising if and which signed language is performed (Moryossef et al., 2020) and sign language recognition (SLR) (Koller, 2020), i.e. the identification of signs either in isolation or in continuous speech. Other tasks concern the translation from signed to spoken (or written) (Camgoz et al., 2018) language or the production of signs from text (Rastgoo et al., 2021). With the recent success of deep learning-based approaches in computer vision (CV), as well as advancements in —from the CV perspective—related tasks of action and gesture recognition (Asadi-Aghbolaghi et al., 2017), SLP is gaining more attention in the CV community (Zheng et al., 2017).

Due to the complexity of the tasks, some recent approaches to various SLP tasks implicitly rely on *phonological* features (Tornay, 2021; Metaxas et al., 2018; Gebre et al., 2013; Tavella et al., 2021). Surprisingly, however, little work has been carried out on explicitly modelling the phonology of signed languages. This presents a timely opportunity to investigate signed languages from the perspective of computational linguistics (Yin et al., 2021). In the context of signed languages, phonology typically distinguishes between manual features, such as usage, position and movement of hands and fingers, and non-manual features, such as facial expressions. Sign language phonology is a matured field with well-developed theoretical frameworks (Liddell and Johnson, 1989; Fenlon et al., 2017; Sandler, 2012). These phonological features, or *phonemes*, are drawn from a fixed inventory of possible configurations which is typically much smaller than the vocabulary of signed languages (Borg and Camilleri, 2020). For example, there is only a limited number of fingers that can be used

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 453 - 463
May 22-27, 2022 ©2022 Association for Computational Linguistics

to perform a sign due to anatomical constraints. Hence, different signs share phonological properties and well performing classifiers can be used to predict those properties for signs unseen during training. This potentially holds even across different languages, because, while different languages may dictate different combinations of phonemes, there are also significant overlaps (Tornay et al., 2020).

Finally, these phonological properties have a strong discriminatory power when determining signs. For example, in ASL-Lex (Caselli et al., 2017), a lexicon which also captures phonology information, the authors report that more than 50% of its 994 described signs have a unique combination of only six phonological properties and more than 80% of the signs share their combination with at most two other signs. By relying on this phonological information from resources such as ASL-Lex, many signs can be uniquely determined. This means that well performing classifiers can leverage this information to predict signs without having encountered them during training. This is a capability that current data-driven approaches to SLR lack by design (Koller, 2020). Thus, in combination, mature approaches to phonology recognition can facilitate the development of sign language resources, for example by providing first-pass silver annotations for new sign languages based on their phonological properties. This is an important task for both documenting low-resource sign languages as well as rapid developing of large-scale datasets, and for fully harnessing data-driven CV approaches.

To spur research in this direction, we extend the preliminary work by Tavella et al. (2021) and introduce the task of Phonological Property Recognition (PPR). More specifically, with this paper, we contribute *(i)* WLASLLex2001, a large-scale, automatically constructed PPR dataset, *(ii)* an analysis of the dataset quality, and *(iii)* an empirical study of the performance of different deep-learning based baselines thereon.

2 Methodology

We address PPR as a classification problem based on features extracted from videos of people speaking SL. Although manual annotation approaches are widely adopted, these are time consuming and require expert knowledge. Instead, we rely on automated dataset construction. On a high level, we cross-reference a large-scale ASL SLR dataset with an ASL Lexicon and annotate videos of signs with their corresponding phonological properties. We then extract skeletal features, by taking advantage of pre-trained deep models from the computer vision community (Rong et al., 2021; Wang et al., 2019). Finally, we train several deep models to classify them as phonological classes.

2.1 Dataset construction

As previously mentioned, ASL-Lex (Caselli et al., 2017) contains phonological features of American Sign Language, such as where the sign is executed, the movement performed by the hand and the number of hands and fingers involved. The latter properties were coded by 3 ASL-versed people. In our work, we are interested in recognising phonological properties from videos of people speaking ASL. Consequently, we aim to construct a dataset, suitable for supervised learning, containing videos labelled with six phonological properties. Specifically, we choose the manual properties with the strongest discriminatory power to determine signs based on their configuration (Caselli et al., 2017):

(i) *flexion:* aperture of the selected fingers of the dominant hand at sign onset,

(ii) *major location:* general location of the dominant hand at sign onset,

(iii) *minor location:* specific location of the dominant hand at sign onset,

(iv) *movement:* the first movement path of the sign,

(v) *selected fingers:* fingers that are moving or are foregrounded during that movement, and

(vi) *sign type:* symmetry of the hands according to Battison (1978).

A detailed description of all the properties is provided in the appendix.

One of the limitations of ASL-Lex is the small number of examples and lack of variety: its first iteration (ASL-Lex 1.0) contains less than 1000 videos, all signed by the same person. While sufficient for educational purposes, these videos are of limited suitability for developing robust classifiers that can capture the diversity of ASL speakers (Yin et al., 2021). To this end, we source videos from WLASL (Li et al., 2020) (Word Level-ASL), one of the largest available SL datasets, featuring more than 2000 glosses demonstrated by over 100 people, for a total of more than 20000 videos. Each sign is performed by at least 3 different signers, which implies greater variability compared to having one gloss performed by only one user. By cross referencing ASL-Lex and WLASL2000 based on corresponding glosses, we can increase the number of samples available to train our models.

Finally, to leverage state of the art SLR architectures that operate over structured input, we enrich each raw video with its extracted keypoints that represent the joints of the speaker. To do so, we use two pretrained models, FrankMocap (Rong et al., 2021) and HRNet (Wang et al., 2019). While these tracking algorithms follow different paradigms, the former extracting 3D coordinates based on a predicted human model and the latter predicting keypoints as coordinates from videos directly, they produce similar outputs. An important distinction is that while FrankMocap estimates the 3D keypoints, HRNet outputs 2D keypoints with associated prediction confidence scores. We use these different models to explore whether different tracking algorithms affect the recognition of phonological classes. We select a subset of features of the upper body, namely: nose, eyes, shoulders, elbows, wrists, thumbs and first/last knuckles of the fingers. These manual features were determined to be the most informative while performing sign language recognition (Jiang et al., 2021b).

Our final dataset, WLASL-Lex2001 (WLASL2000 + ASL-Lex 1.0), is composed of 10017 videos corresponding to 800 glosses, 3D skeletons (x, y, z from FrankMocap and x, y and $score$ from HRNet) labelled with their phonological properties. A characteristic of this dataset is that it follows a long tailed distribution. Due to the nature of language, some phonological properties are more common than others, which means that some classes are more represented than others. On the one hand, the training setup for our models should take this factor into account, but on the other hand, the advantage of training over phonological classes instead of glosses is that different glosses can share phonological classes.

2.2 Models

To estimate the complexity of the dataset, we use the majority-class baseline and the Multi-Layer Perceptron (MLP) as basic deep models. We further use Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) as models capable of capturing the temporal component of videos. As state-of-the-art SLP architectures that have been used to perform SLR, we use the I3D 3D Convolutional Neural Network (Carreira and Zisserman, 2017; Li et al., 2020) able to learn from raw videos, and the Spatio-Temporal Graph Convolutional Network (STGCN) (Jiang et al., 2021b) that captures both spatial and temporal components from the extracted keypoints.

2.3 Experimental Setup

For each phonological property we generate dataset splits and train dedicated models separately. While a multi-class multi-label approach could achieve higher scores, by relying on potential interdependencies of different properties, we chose to model the properties in isolation, to disentangle the factors that affect the learnability of each property. From now on, when we mention the *dataset*, we refer to an instance of the WLASL-Lex 2001 dataset, where labels are the values of a single phonological class.

We make this distinction because we produce six different train, validation and test splits (with a $70 : 15 : 15$ ratio) stratifying on the corresponding phonological property (*Phoneme*). By doing so, we make sure that *(a)* all splits contain all possible labels for a classification target (i.e. phonological property) and *(b)* follow the same distribution. Since we source the videos from WLASL, we have multiple videos representing each gloss, therefore, randomly splitting our data will result in the fact that glosses in the test set might appear in the training set as well, signed by a different speaker. Thus, to investigate how well the models can predict properties on unseen glosses, we also produce label-stratified splits on gloss-level (*Gloss*), such that videos of glosses in the validation and test set do not appear in training data and vice versa. Thus, to summarise, experiments in the *Phoneme* setting aim to evaluate the capability to recognise phonological properties of signs that were already encountered in the training data, but are performed by a different speaker in the test set. Conversely, experiments in the *Gloss* setting aim to evaluate the capability to recognise phonological properties of signs completely *unseen during training*.

We use an I3D model that has been pre-trained on Kinetics-400 (Carreira and Zisserman, 2017)

		FLEXION		MAJLOCATION		MINLOCATION		MOVEMENT		FINGERS		SIGNTYPE	
		A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$
Phoneme	Baseline	50.3	11.1	34.4	20.0	33.9	3.1	35.5	16.7	48.2	11.1	39.3	20
	MLP$_H$	44.1 ± 2.5	11.1	70.3 ± 2.3	64.0	51.6 ± 2.5	28.2	34.5 ± 2.4	18.7	59.4 ± 2.5	25.0	73.9 ± 2.2	52.6
	MLP$_F$	50.3 ± 2.5	11.1	57.8 ± 2.5	46.8	34.3 ± 2.4	9.1	34.3 ± 2.4	18.7	43.4 ± 2.5	12.9	67.0 ± 2.4	42.8
	RNN$_H$	49.0 ± 2.5	30.0	75.8 ± 2.2	72.4	64.3 ± 2.4	46.0	35.1 ± 2.4	29.5	71.0 ± 2.3	46.5	78.7 ± 2.1	58.8
	RNN$_F$	50.3 ± 2.5	11.1	64.6 ± 2.4	54.2	30.3 ± 2.3	4.0	35.4 ± 2.4	18.1	46.5 ± 2.5	12.4	70.9 ± 2.3	46.8
	STGCN$_H$	**62.3 ± 2.4**	**45.0**	**83.2 ± 1.9**	**78.6**	**74.5 ± 2.2**	**63.5**	**63.6 ± 2.4**	**58.2**	**73.8 ± 2.2**	**56.0**	**84.5 ± 1.8**	**69.6**
	STGCN$_F$	43.4 ± 2.5	20.8	70.5 ± 2.3	62.1	53.0 ± 2.5	40.0	45.7 ± 2.5	37.8	63.1 ± 2.4	32.8	73.0 ± 2.2	53.1
	3DCNN	46.5 ± 2.5	13.2	64.3 ± 2.4	55.2	42.3 ± 2.5	18.6	32.9 ± 2.4	20.8	47.5 ± 2.5	14.5	69.5 ± 2.3	44.8
Gloss	Baseline	**53.1**	11.1	35.7	20.0	42.0	5.0	35.2	16.7	47.4	12.5	38.3	20.0
	MLP$_H$	44.6 ± 2.5	15.5	68.1 ± 2.3	56.6	47.3 ± 2.5	19.7	28.4 ± 2.2	19.8	56.2 ± 2.5	22.9	75.3 ± 2.2	50.7
	MLP$_F$	52.8 ± 2.5	11.1	56.6 ± 2.5	42.9	38.3 ± 2.4	10.7	37.1 ± 2.4	21.7	39.3 ± 2.5	12.5	68.4 ± 2.4	41.2
	RNN$_H$	39.6 ± 2.5	18.0	72.8 ± 2.2	67.3	49.3 ± 2.5	26.3	32.2 ± 2.3	24.9	60.7 ± 2.5	32.5	75.4 ± 2.2	53.5
	RNN$_F$	53.0 ± 2.5	11.1	64.1 ± 2.4	52.6	44.4 ± 2.4	17.8	36.7 ± 2.4	20.1	27.3 ± 2.3	12.7	72.0 ± 2.3	46.9
	STGCN$_H$	49.1 ± 2.5	**21.6**	**77.3 ± 2.1**	**70.0**	**55.1 ± 2.4**	**32.7**	**52.5 ± 2.5**	**46.5**	**65.7 ± 2.4**	**34.4**	**76.6 ± 2.1**	**54.4**
	STGCN$_F$	39.0 ± 2.5	14.4	66.7 ± 2.3	60.1	45.1 ± 2.4	21.1	43.1 ± 2.5	34.9	60.0 ± 2.5	29.2	71.3 ± 2.3	47.5
	3DCNN	46.0 ± 2.5	12.8	64.9 ± 2.4	52.0	10.8 ± 1.5	13.6	32.0 ± 2.3	19.3	45.9 ± 2.5	14.7	71.6 ± 2.3	46.3

Table 1: Accuracy (A.) and per-class averaged accuracy ($\overline{A}$) of various models on the test sets of the six tasks. For accuracy, we report the error margin as a confidence interval at $\alpha = 0.05$ using asymptotic normal approximation. We omit error margins for balanced accuracy as the low number of classes results in a small sample size. Additional performance measures are reported in the appendix.

and fine-tune it on raw videos from our datasets. The other models are trained from scratch using keypoints as input. We fix the length of all input to 150 frames, longer sequences are truncated while shorter sequences are looped to reach the fixed length. We select the best performing model based on performance on the validation set and for the final test set performance we train the models on both train and validation sets. For more details on model selection, consult the appendix. We measure both accuracy, to investigate how well models perform in general, and class-balanced accuracy to take into account how well they are able to model different classes of the phonological properties.

3 Results and discussion

The upper half of Table 1 presents the results for the six dataset splits for the *Phoneme* setting, where glosses in test data could have appeared in training data as well. The poor performance of the simple MLP architecture suggests that the tasks are in fact challenging and do not exhibit easily exploitable regularities. Due to its simplicity, it is barely able to reach the baseline for some properties (34% vs. 35% and 44% vs. 50% for *movement* and *flexion* respectively). In particular, MLP classifying based on FrankMocap (MLP$_F$) output is often the worst performing combination. Conversely, STGCN using HRNet output (STGCN$_H$) outperforms other models on all six tasks. In some cases, for example when predicting *movement* or *flexion*, it is the only model which significantly surpasses the majority class baseline. This superior performance is expected, as this specific combination of the STGCN operating over HRNet-extracted keypoints has been shown to be the largest contributor to the SLR performance on the WLASL2000 dataset (Jiang et al., 2021a).

Models that operate over structured input often outperform the 3D CNN, demonstrating the utility of additional information provided by the skeleton features. The results also suggest that models using the HRNet skeleton output outperform those who use FrankMocap, possibly due to the confidence scores produced by HRNet and associated with the coordinates. This difference in performance suggests to conduct a more rigorous study to investigate the impact of different feature extraction methods as a possible future research direction.

The lower half of Table 1 shows the performance of models to predict the phonological properties of unseen glosses (*Gloss*). The performance of all tasks and all models deteriorates, suggesting that their success is partly derived from exploiting the similarities between glosses that appear in training and test data. However, the best model, STGCN$_H$, performs comparably to the *Phoneme*-split, with a drop of less than 10 accuracy points for five of the six tasks.

Often, crowd sourced (Polonio et al., 2018) or automatically constructed datasets such as ours, have a performance ceiling, possibly due to incorrectly assigned ground truth labels or low quality of input data (Chen et al., 2016; Schlegel et al., 2020). To investigate the former, we measure the agreement on videos that all models misclassify using Fleiss' κ. Intuitively, if models consistently agree on a label different than the ground truth, the ground truth label might be wrong. We find that averaged across the six tasks, the agreement is negligible: 0.09 ± 0.06 and 0.11 ± 0.09 for *Phoneme* and *Gloss* split, respectively.

Similarly, for the latter, if all models consistently fail to assign any correct label for a given video (e.g. all models err on a video appearing in the test sets of *movement* and *flexion*), this can hint at low quality of the input, making it impossible to predict anything correctly. We find that this is not the case with WLASL-LEX2001, as videos appearing in test sets of different tasks tend to have a low mutual misclassification rate: 1% and 0.7% of videos appearing in test sets of two and three tasks were misclassified by all models for all associated tasks for the *Phoneme* split. For the *Gloss* split the numbers are 3 and 0% for two and three tasks, respectively. Together, these observations suggest that the models presented in this paper are unlikely to reach the performance ceiling on WLASL-Lex2001 and more advanced approaches could obtain even higher accuracy scores.

4 Conclusion

In this paper, we discuss the task of Phonological Property Recognition (PPR). We automatically construct a dataset for the task featuring six phonological properties and analyse it extensively. We find that there is potential for improvement over our presented data-driven baseline approaches. Researchers pursuing this direction can focus on developing better-performing models, for example by relying on jointly learning all properties, as labels for different properties can be mutually dependent.

Another possible avenue is to investigate the feasibility of using PRR to perform *tokenisation* of continuous sign language speech, by decomposing it into multiple phonemes, which is identified as one of the big challenges of SLP (Yin et al., 2021).

Acknowledgements

The authors would like to acknowledge the use of the Computational Shared Facility at The University of Manchester. The work was partially supported by the UKRI TAS Node on Trust, the US Air Force project THRIVE++ and the H2020 projects TRAINCREASE, eLADDA and PERSEO.

References

Maryam Asadi-Aghbolaghi, Albert Clapes, Marco Bellantonio, Hugo Jair Escalante, Victor Ponce-Lopez, Xavier Baro, Isabelle Guyon, Shohreh Kasaei, and Sergio Escalera. 2017. A Survey on Deep Learning Based Approaches for Action and Gesture Recognition in Image Sequences. *Proceedings - 12th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2017 - 1st International Workshop on Adaptive Shot Learning for Gesture Understanding and Production, ASL4GUP 2017, Biometrics in the Wild, Bwild 2017, Heteroge*, pages 476–483.

Robbin Battison. 1978. Lexical borrowing in american sign language.

Mark Borg and Kenneth P. Camilleri. 2020. Phonologically-Meaningful Subunits for Deep Learning-Based Sign Language Recognition. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12536 LNCS:199–217.

Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. 2018. Neural Sign Language Translation.

João Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733.

Naomi K. Caselli, Zed Sevcikova Sehyr, Ariel M. Cohen-Goldberg, and Karen Emmorey. 2017. Asllex: A lexical database of american sign language. *Behavior Research Methods*, 49(2):784–801.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 4, pages 2358–2367, Stroudsburg, PA, USA. Association for Computational Linguistics.

J Fenlon, Kearsy A Cormier, and Diane Brentari. 2017. Sign language phonology. In *Routledge Handbook of Phonological Theory*.

Binyam Gebrekidan Gebre, Peter Wittenburg, and Tom Heskes. 2013. Automatic sign language identification. *2013 IEEE International Conference on Image Processing, ICIP 2013 - Proceedings*, pages 2626–2630.

Songyao Jiang, Bin Sun, Lichen Wang, Yue Bai, Kunpeng Li, and Yun Fu. 2021a. Sign Language Recognition via Skeleton-Aware Multi-Model Ensemble.

Songyao Jiang, Bin Sun, Lichen Wang, Yue Bai, Kunpeng Li, and Yun Fu. 2021b. Skeleton Aware Multimodal Sign Language Recognition. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 3408–3418.

Oscar Koller. 2020. Quantitative Survey of the State of the Art in Sign Language Recognition.

Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. 2020. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1459–1469.

Scott K. Liddell and Robert E. Johnson. 1989. American Sign Language: The Phonological Base. *Sign Language Studies*, 1064(1):195–277.

B.W. Matthews. 1975. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451.

Dimitris Metaxas, Mark Dilsizian, and Carol Neidle. 2018. Scalable ASL sign recognition using model-based machine learning and linguistically annotated corpora.

Amit Moryossef, Ioannis Tsochantaridis, Roee Aharoni, Sarah Ebling, and Srini Narayanan. 2020. Real-Time Sign Language Detection Using Human Pose Estimation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12536 LNCS:237–248.

Davide Polonio, Federico Tavella, Marco Zanella, and Armir Bujari. 2018. Ghio-ca: An android application for automatic image classification. In *Smart Objects and Technologies for Social Good*, pages 248–257, Cham. Springer International Publishing.

Razieh Rastgoo, Kourosh Kiani, and Sergio Escalera. 2021. Sign Language Recognition: A Deep Survey. *Expert Systems with Applications*, 164:113794.

Yu Rong, Takaaki Shiratori, and Hanbyul Joo. 2021. Frankmocap: A monocular 3d whole-body pose estimation system via regression and integration. In *IEEE International Conference on Computer Vision Workshops*.

Wendy Sandler. 2012. The Phonological Organization of Sign Languages. *Language and Linguistics Compass*, 6(3):162–182.

Viktor Schlegel, Marco Valentino, André Andre Freitas, Goran Nenadic, and Riza Batista-Navarro. 2020. A Framework for Evaluation of Machine Reading Comprehension Gold Standards. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5359–5369, Marseille, France. European Language Resources Association.

Federico Tavella, Aphrodite Galata, and Angelo Cangelosi. 2021. Phonology recognition in american sign language.

Sandrine Tornay. 2021. *Explainable Phonology-based Approach for Sign Language Recognition and Assessment*. Ph.D. thesis, Lausanne, EPFL.

Sandrine Tornay, Marzieh Razavi, and Mathew Magimai.-Doss. 2020. Towards Multilingual Sign Language Recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6304–6308. IEEE.

Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. 2019. Deep high-resolution representation learning for visual recognition. *TPAMI*.

Kayo Yin, Amit Moryossef, Julie Hochgesang, Yoav Goldberg, and Malihe Alikhani. 2021. Including Signed Languages in Natural Language Processing. pages 7347–7360.

Lihong Zheng, Bin Liang, and Ailian Jiang. 2017. Recent Advances of Deep Learning for Sign Language Recognition. *DICTA 2017 - 2017 International Conference on Digital Image Computing: Techniques and Applications*, 2017-Decem:1–7.

A Hyperparameters optimization

Table 2 contains all the hyperparameters explored during our experiment over each different model. The best model is the one that maximises the Matthew's correlation coefficient

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

with TP, TN, FP, FN being true/false positive/negative. For the STGCN we use hyperparameters chosen by Jiang et al. (2021a), because initial experiments on our data showed a difference of at most 2% accuracy, which is within the uncertainty estimate. To find the optimal hyperparameters for the other models, we perform Bayesian optimisation over a pre-defined set. We maximise Matthews correlation coefficient (MCC) (Matthews, 1975) on the validation sets of all six tasks. We choose MCC as it provides a good trade-off between overall and class-level accuracy which is necessary due to the unbalance inherently present in our dataset.

Model	Parameters
MLP	number of layers hidden dimension dropout learning rate scheduler step size gamma
RNN	number of RNN layers RNN hidden dimension RNN dropout
STGCN	learning rate number of groups block size, window size scheduler step size dropout warmup epochs
3D CNN	dropout learning rate gamma scheduler step size window size

Table 2: Set of explored hyperparameters for each different model

B Seed dependency

Table 3 illustrates the performance on the test set for each model with respect to chance as measured by training 5 models from different random seeds. The performance difference is negligible suggesting that model training is largely stable with regard to chance.

Model	Accuracy
MLP	74.39 ± 0.35
RNN	79.12 ± 0.46
STGCN	84.12 ± 0.29
3D CNN	69.23 ± 0.93

Table 3: Mean and standard deviation of accuracy of all architectures trained with the HRNet output, measured on the SIGNTYPE test set and averaged over 5 different random seeds. Results for the 3D CNN are obtained from the validation set.

C Phonological classes description

Tables 4 to 9 describe in detail the meaning of values for all the phonological classes according to ASL-Lex (Caselli et al., 2017).

The cardinality is calculated on WLASL-Lex, which is why some classes that are in ASL-Lex are not represented (i.e., cardinality equal to 0).

D Additional results

Table 10 illustrates additional results for several different metrics. In particular, we report micro- and macro precision/recall and Matthews correlation coefficient. These metrics help to give a better understanding of the classification results, as they are affected more by data imbalance when compared to accuracy.

Value	Definition	Cardinality
imrp	index, middle, ring, pinky finger	4824
imr	index, middle, ring finger	95
mrp	middle, ring, pinky finger	28
im	index, middle finger	1296
ip	index, pinky finger	51
mr	middle, ring finger	0
mp	middle, pinky finger	0
rp	ring, pinky finger	0
i	index finger	2547
m	middle finger	259
r	ring finger	0
p	pinky	407
thumb	thumb	510

Table 4: Values and relative definitions for selected fingers

Value	Definition	Cardinality
Head	Sign is produced on or near the head	3137
Arm	Sign is produced on or near the arm	219
Body	Sign is produced on or near the trunk	1019
Hand	Sign is produced on or near the non-dominant hand	2194
Neutral	Sign is not produced in another location on the body	3448
Other	Sign is produced in another unspecified location on the body	0

Table 5: Values and relative definitions for major location

Value	Definition	Cardinality
1	Fully open: no joints of selected fingers are flexed	5037
2	Bent (closed): non-base joints are flexed	693
3	Flat-open: base joints flexed less than 90 degrees	909
4	Flat-closed: base joints flexed equal to or more that 90 degrees	507
5	Curved open: base and non-base joints flexed without contact	1130
6	Curved closed: base and non-base joints flexed with contact	642
7	Fully closed: base and non-base joints fully flexed	795
Stacked	Stacked: Flexion of selected fingers differs	123
Crossed	Crossed	181

Table 6: Values and relative definitions for flexion

Value	**Definition**	**Cardinality**
HeadTop	Sign is produced on top of the head	20
Forehead	Sign is produced at the forehead	246
Eye	Sign is produced near the eye	616
CheekNose	Sign is produced on the cheek or nose	511
UpperLip	Sign is produced on the upper lip	53
Mouth	Sign is produced on the mouth	431
Chin	Sign is produced on the chin	717
UnderChin	Sign is produced under the chin	74
UpperArm	Sign is produced on the upper arm	39
ElbowFront	Sign is produced in the crook of the elbow	0
ElbowBack	Sign is produced on the outside of the elbow	13
ForearmBack	Sign is produced on the outside of the forearm	32
ForearmFront	Sign is produced on the inside of the forearm	10
ForearmUlnar	Sign is produced on the ulnar side of the forearm	56
WristBack	Sign is produced on the back of the wriset	23
WristFront	Sign is produced on the front of the wrist	0
Neck	Sign is produced on the neck	68
Shoulder	Sign is produced on the shoulder	101
Clavicle	Sign is produced on the clavicle	419
TorsoTop	Sign is produced in the upper third of the torso	0
TorsoMid	Sign is produced in the middle third of the torso	0
TorsoBottom	Sign is produced in the bottom third of the torso	19
Waist	Sign is produced at the waist	34
Hips	Sign is produced on the hips	59
Palm	Sign is produced on the plam of the non-dominant hand	925
FingerFront	Sign is produced on the front of the fingers of the non-dominant hand	99
PalmBack	Sign is produced on the back of the palm of the non-dominant hand	218
FingerBack	Sign is produced on the back of the fingers of the non-dominant hand	186
FingerRadial	Sign is produced on the radial side of the non-dominant hand	410
FingerUlnar	Sign is produced on the ulnar side of the non-dominant hand	40
FingerTip	Sign is produced on the tip of the fingers of the non-dominant hand	158
Heel	Sign is produced on the heel of the non-dominant hand	88
Other	Sign is produced in an unspecified location on the body	707
Neutral	Sign is not produced on or near the body	3390

Table 7: Values and relative definitions for minor location

Value	Definition	Cardinality
One Handed	Sign only recruits one hand	3939
Symmetrical Or Alternating	Sign recruits both hands Phonological specifications for both hands are identical Movement of both hands is either symmetrical or alternating	3358
Asymmetrical Same Handshape	Sign recruits both hands Only the dominant hand moves The location and orientation of the hands may differ, but the other specifications of handshape are the same Non-Dominant hand must be an unmarked handshape (B A S 1 C O 5)	938
Asymmetrical Different Handshape	Sign recruits both hands Only the dominant hand moves The location and orientation of the hands may differ, and the other specifications of handshape are not the same Non-Dominant hand must be an unmarked handshape (B A S 1 C O 5)	1639
Other	Sign violates Battison's Symmetry and Dominance Conditions	143

Table 8: Values and relative definitions for sign type

Value	Definition	Cardinality
Straight	Straight movement of the dominant hand through xyz space	1938
Curved	Single arc movement of the dominant hand through xyz space Hands may or may not make contact with multiple locations	1255
BackAndForth	Sequence of more than one straight or curved movements	3549
Circular	Circular movement of the dominant hand through space Rotation alone does not constitute a circular movement	1129
None	Entire sign (or first free morpheme) does not have a path movement	1748
Other	Sign has another unspecified path movement	398

Table 9: Values and relative definitions for movement

		Flexion					**MajLocation**					**MinLocation**					**Movement**					**Fingers**					**Signtype**				
		P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC
Phoneme	Baseline	50.3	5.59	50.3	11.11	0.0	34.4	6.88	34.4	20.0	0.0	33.87	1.06	33.87	3.12	0.0	35.46	5.91	35.46	16.67	0.0	48.17	5.35	48.17	11.11	0.0	39.32	7.86	39.32	20.0	0.0
	MLP$_H$	44.1	24.5	44.1	20.7	14.6	70.3	65.8	70.3	64.0	58.9	51.6	37.3	51.6	28.2	41.6	34.5	28.0	34.5	26.9	13.9	59.5	29.6	59.5	25.0	37.7	73.9	54.1	73.9	52.6	62.5
	MLP$_F$	50.3	5.6	50.3	11.1	0.9	57.8	52.3	57.8	46.8	41.2	34.3	13.9	34.3	9.1	17.9	34.3	13.1	34.3	18.7	5.7	43.4	17.5	43.4	12.9	4.6	67.1	38.1	67.1	42.8	52.7
	RNN$_H$	49.0	32.1	49.0	30.0	25.4	75.8	75.2	75.8	72.4	66.4	64.3	54.3	64.3	46.0	57.4	35.1	30.1	35.1	29.5	15.9	71.0	53.3	71.0	46.5	56.6	78.7	61.2	78.7	58.8	69.4
	RNN$_F$	50.3	5.6	50.3	11.1	0.0	63.9	56.7	63.9	52.2	50.1	30.3	4.7	30.3	4.0	4.9	35.4	21.4	35.4	18.1	5.2	46.5	9.2	46.5	12.4	8.5	70.9	60.6	70.9	46.8	58.3
	GTN$_H$	62.4	55.4	62.4	45.0	43.9	83.2	80.6	83.2	78.6	76.8	74.5	66.7	74.5	63.5	69.8	63.6	62.1	63.6	58.2	52.7	73.8	71.7	73.8	56.0	61.1	84.5	74.9	84.5	69.6	77.7
	GTN$_F$	43.4	23.6	43.4	20.8	15.3	70.5	66.4	70.5	62.1	58.9	53.0	43.9	53.0	40.0	43.8	45.7	40.8	45.7	37.8	28.6	63.1	39.0	63.1	32.8	44.3	73.0	56.8	73.0	53.1	61.1
	3DCNN	46.5	17.8	46.5	13.2	5.4	64.3	57.2	64.3	55.2	50.3	42.3	22.8	42.3	18.6	29.1	32.9	23.4	32.9	20.8	7.5	47.5	17.8	47.5	14.5	14.6	69.5	44.9	69.5	44.8	55.6
Gloss	Baseline	53.03	5.89	53.03	11.11	0.0	35.69	7.14	35.69	20.0	0.0	42.03	2.1	42.03	5.0	0.0	35.21	5.87	35.21	16.67	0.0	47.38	5.92	47.38	12.5	0.0	38.28	7.66	38.28	20.0	0.0
	MLP$_H$	44.6	18.6	44.6	15.5	8.3	68.1	62.0	68.1	56.6	55.5	47.3	16.8	47.3	13.1	32.5	28.4	20.4	28.4	19.8	4.9	56.2	21.4	56.2	20.3	32.0	75.3	50.6	75.3	50.7	64.3
	MLP$_F$	52.8	5.9	52.8	11.1	−2.1	56.7	46.2	56.7	42.9	39.5	38.3	11.7	38.3	7.9	18.1	37.1	15.9	37.1	21.7	12.5	39.3	10.4	39.3	11.1	0.4	68.4	37.7	68.4	41.2	54.3
	RNN$_H$	39.6	19.8	39.6	18.0	10.9	72.8	68.0	72.8	67.3	62.4	49.3	19.6	49.3	17.5	36.7	32.2	25.7	32.2	24.9	11.3	60.7	36.9	60.7	32.5	40.3	75.4	55.0	75.4	53.5	64.6
	RNN$_F$	53.0	5.9	53.0	11.1	0.0	64.1	57.3	64.1	52.6	50.5	44.4	15.1	44.4	12.3	27.9	36.7	11.2	36.7	20.1	10.0	27.3	10.6	27.3	12.7	3.0	72.0	41.3	72.0	46.9	60.4
	GTN$_H$	49.1	25.6	49.1	21.6	18.9	77.3	72.1	77.3	70.0	68.6	55.1	25.1	55.1	23.3	43.4	52.5	49.4	52.5	46.5	38.0	65.7	37.2	65.7	30.6	47.8	76.6	54.9	76.6	54.4	66.2
	GTN$_F$	39.0	15.1	39.0	14.4	4.7	66.7	63.2	66.7	60.1	53.9	45.1	15.7	45.1	13.2	31.1	43.1	36.0	43.1	34.9	25.8	60.0	32.5	60.0	29.2	39.4	71.3	47.6	71.3	47.5	58.5
	3DCNN	46.0	12.0	46.0	12.8	4.5	65.0	57.5	65.0	52.0	51.8	10.8	12.0	10.8	9.7	9.5	32.0	18.7	32.0	19.3	6.0	45.9	15.1	45.9	14.7	10.7	71.6	46.3	71.6	46.3	58.7

Table 10: Micro-averaged (μ), macro-averaged (M) precision (P) and recall (R) and Matthews correlation coefficient (MCC) of various models on the test sets of the six tasks. We omit error margins as the low number of classes results in a small sample size.

Investigating person-specific errors in chat-oriented dialogue systems

Koh Mitsuda[†], **Ryuichiro Higashinaka**[†], **Tingxuan Li**[*], **Sen Yoshida**[†]
[†]NTT Corporation, Japan
[*]University of Tsukuba, Japan
{koh.mitsuda.td, ryuichiro.higashinaka.tp,
sen.yoshida.tu}@hco.ntt.co.jp, s2120816@s.tsukuba.ac.jp

Abstract

Creating chatbots to behave like real people is important in terms of believability. Errors in general chatbots and chatbots that follow a rough persona have been studied, but those in chatbots that behave like real people have not been thoroughly investigated. We collected a large amount of user interactions of a generation-based chatbot trained from large-scale dialogue data of a specific character, i.e., "target person" and analyzed errors related to that person. We found that person-specific errors can be divided into two types: errors in attributes and those in relations, each of which can be divided into two levels: self and other. The correspondence with an existing taxonomy of errors was also investigated, and person-specific errors that should be addressed in the future were clarified.

1 Introduction

Creating chatbots to behave like real people is important in terms of believability (Traum et al., 2015; Higashinaka et al., 2018). Errors in general chatbots (Higashinaka et al., 2021) and chatbots that follow a rough persona (Li et al., 2016; Zhang et al., 2018; Zhou et al., 2020; Inoue et al., 2020; Song et al., 2020; Roller et al., 2020) have been studied, but those in chatbots that behave like real people have not been thoroughly investigated.

We analyzed dialogue data between a chatbot that imitates a certain person and users to identify "errors related to the target person" (hereafter referred to as **person-specific errors**). We collected a large amount of dialogue data between users and the latest generation-based chatbot trained with a large amount of dialogue data of the target person and analyzed the errors. The results indicate that person-specific errors can be divided into two types: errors in attributes and those in relations, each of which can be divided into two levels: self and other. The correspondence with the existing taxonomy of errors was also investigated, and errors that should be addressed in the future were clarified.

2 Dialogue data collection

We used a chatbot that imitates a specific person. By making the chatbot available to the public, we collected dialogue data from a large number of users.

2.1 Chatbot

In our previous study, we collected a large amount of dialogue data on a target person and created a chatbot by fine-tuning a pre-trained encoder-decoder Transformer model (Mitsuda et al., 2021). The specific character (i.e., target person) was Amadeus Kurisu, a character in a famous Japanese video game (STEINS;GATE). We used a role-play-based question-answering (QA) scheme proposed by Higashinaka et al. (2018), in which fans of a character provided questions and answers by role-playing to collect the dialogue data on that character. We collected a large amount of QA pairs (44,805) from the fans. To add multi-turn dialogues, we additionally created 4,500 dialogues (24,750 utterances) by manually extending the collected QA pairs.

As a pre-trained dialogue model, we used the Japanese version of BlenderBot (Japanese-dialog-transformers[1]) created by Sugiyama et al. (2021). They pre-trained the encoder-decoder Transformer using 2.1B dialogues crawled from Twitter in Japanese then fine-tuned the model with the corpora including the Japanese version of PersonaChat (Zhang et al., 2018) and EmpatheticDialogues (Rashkin et al., 2018). We created the chatbot for Kurisu by further fine-tuning the model with the collected QA pairs and extended dialogue

[*] Work carried out during internship at NTT.

[1] https://github.com/nttcslab/japanese-dialog-transformers

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 464 - 469
May 22-27, 2022 ©2022 Association for Computational Linguistics

data. To evaluate the fine-tuned model, 20 workers interacted with the chatbot by performing 15-turn dialogues (a turn corresponds to a user utterance and chatbot utterance: hereafter, system utterance) three times. The subjective evaluation results on naturalness, characterness, and informativeness were 3.87, 3.90, and 3.58, respectively (on a 5-point Likert scale).

2.2 Large-scale user study

The chatbot described in the previous section was made public on the Internet, and the dialogues between a large number of users, mostly the fans of Kurisu, and the chatbot were collected. The chatbot was accessible using the direct message function of Twitter for three days. After users agreed to the terms of usage, they could interact with the chatbot. Users could stop the dialogue at any time or interact with it as much as they wanted during the period. At the end of the study, a user questionnaire (on a 5-point Likert scale) was sent out by direct message to the users to evaluate user satisfaction. Note that the users were not paid for their participation.

We were able to collect the logs of 1,170 user interactions with the chatbot. The total number of user utterances was 80,608, and the average number of utterances for each user was 68.9, indicating that the users used the chatbot for a relatively long time. The average user-satisfaction rating was 4.59 (63.6% response rate), which we believe is very high.

3 Error analysis

To extract system utterances causing person-specific errors from the data, we collected four types of information: dialogue breakdown labels, comments on the reasons for the breakdown (Higashinaka et al., 2015), flags indicating whether the comments were about the person in question, and error types in chat-oriented dialogue systems (Higashinaka et al., 2021). We first collected the dialogue breakdown labels and comments on their reasons. If the comments contained keywords related to Kurisu, we considered the system utterances with those comments as indicating person-specific errors and extracted the comments for analysis. We also annotated system utterances with the error types in chat-oriented dialogue systems for investigating the correspondence between the existing taxonomy of errors and person-specific errors.

No. of system utterances	10,611
No. of users (dialogues)	385
No. of workers for dialogue breakdown annotation	5
No. of annotated dialogue breakdown labels	53,055
No. of not breakdowns (NBs)	47,200 (89.0%)
No. of possible breakdowns (PBs)	3,678 (6.9%)
No. of breakdowns (Bs)	2,177 (4.1%)
No. of NB utterances (by majority)	9,794 (92.3%)
No. of PB/B utterances (by majority)	817 (7.7%)

Table 1: Statistics of annotated dialogue breakdown labels

3.1 Dialogue breakdown annotation

We sampled and annotated 13% (= 10,611/80,608) of the data due to the limited annotation resources. The sampled system utterances were annotated with the three types of breakdown labels (Higashinaka et al., 2015) of “not a breakdown (NB)”, “possible breakdown (PB)”, and “breakdown (B)”. Five crowdworkers who had sufficient knowledge of Kurisu annotated these labels to the system utterances independently. The workers were instructed to provide comments to describe the errors that led to the breakdowns.

Table 1 shows the annotation results of the dialogue breakdown labels. The percentage of NBs was 89%, indicating that the dialogue was successful in the majority of cases. The inter-annotator agreement rate was 0.23 for the Fleiss' kappa when NB/PB/B were treated separately and 0.30 when PB/B were merged, which was at the same level as in the study by Higashinaka et al. (2015), which we consider reasonable due to the subjective nature of the task. In the following analysis, the system utterances in which more than half the workers marked PB or B were considered for error analysis. The number of such utterances was 817 (7.7%). The error comments (2,846) given to these utterances were also retrieved for analysis.

3.2 Annotation of error types and person-related flags

Two types of information were assigned to the erroneous system utterances and error comments. The first is the error types in chat-oriented dialogue systems (Higashinaka et al., 2021). This labeling was done by an in-house expert worker. The second is a flag indicating whether person-related keywords are present in the error comment. By referring to the resources of Kurisu, we manually created a lexicon of that character.

Error types for chatbots	All	Person-specific errors
(I1) Uninterpretable	9 (1.1%)	0 (0.0%)
(I2) Grammatical error	3 (0.4%)	0 (0.0%)
(I3) Semantic error	10 (1.2%)	3 (30.0%)
(I4) Wrong information	81 (9.8%)	43 (53.1%)
(I5) Ignore question	66 (8.0%)	7 (10.6%)
(I6) Ignore request	10 (1.2%)	1 (10.6%)
(I7) Ignore proposal	0 (0.0%)	0 (–)
(I8) Ignore greeting	0 (0.0%)	0 (–)
(I9) Ignore expectation	119 (14.4%)	30 (25.2%)
(I10) Unclear intention	266 (32.2%)	45 (16.9%)
(I11) Topic transition error	15 (1.8%)	3 (20.0%)
(I12) Lack of info. error	6 (0.7%)	1 (16.7%)
(I13) Self-contradiction	62 (7.5%)	14 (22.6%)
(I14) Contradiction	23 (2.8%)	2 (8.7%)
(I15) Repetition	142 (17.2%)	19 (13.4%)
(I16) Lack of sociality	5 (0.6%)	0 (0.0%)
(I17) Lack of common sense	0 (0.0%)	0 (–)
Total	817 (100%)	168 (20.1%)

Table 2: Results of labeling each error-containing utterance with error type (Higashinaka et al., 2021) and whether it was person-specific error. Numbers in each column indicate number of utterances, and those in parentheses indicate percentage of total number of utterances containing errors.

The size of the lexicon was 53 words. If a word in the lexicon was included in each comment, it was flagged as that related to Kurisu. For example, the lexicon includes Kurisu, Mayuri (the name of Kurisu's friend), @channel (the website that Kurisu is familiar with), and Akihabara (the place where Kurisu resides). o

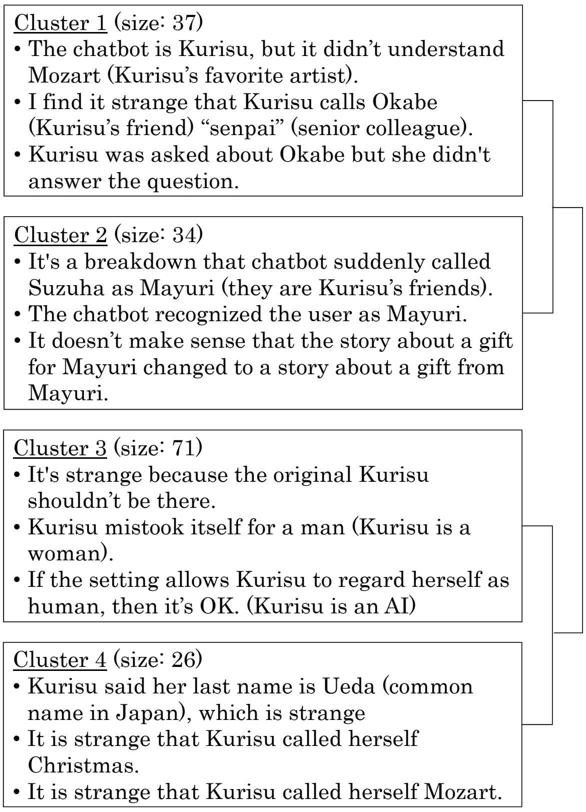
Table 2 shows the annotation results of the error types and number of person-specific errors for each type. In the total number of dialogue breakdowns (817), 168 (20.1%) were caused by person-specific errors, and more than half (53.1%) of the utterances in (I4) Wrong information were person-specific errors.

4 Person-specific error analysis

We automatically clustered the error comments related to the target person and investigated the characteristics the person-specific errors.

4.1 Clustering person-specific errors

We used hierarchical clustering by using bag-of-words as the clustering method. The 168 comments annotated to the 168 person-specific errors shown in Table 2 were used for clustering. A Japanese morphological analyzer JTAG (Fuchi and Takagi, 1998) was used. Low-frequency words (those appearing less than three times in the 168 comments) were excluded. The vector of each comment was normalized for the clustering. Single-linkage clustering, Ward's method, and squared Euclidean distance were specified as clustering parameters. The number of clusters was set so that the size of each cluster would be at least 10% of the total comments.



Cluster 1 (size: 37)
- The chatbot is Kurisu, but it didn't understand Mozart (Kurisu's favorite artist).
- I find it strange that Kurisu calls Okabe (Kurisu's friend) "senpai" (senior colleague).
- Kurisu was asked about Okabe but she didn't answer the question.

Cluster 2 (size: 34)
- It's a breakdown that chatbot suddenly called Suzuha as Mayuri (they are Kurisu's friends).
- The chatbot recognized the user as Mayuri.
- It doesn't make sense that the story about a gift for Mayuri changed to a story about a gift from Mayuri.

Cluster 3 (size: 71)
- It's strange because the original Kurisu shouldn't be there.
- Kurisu mistook itself for a man (Kurisu is a woman).
- If the setting allows Kurisu to regard herself as human, then it's OK. (Kurisu is an AI)

Cluster 4 (size: 26)
- Kurisu said her last name is Ueda (common name in Japan), which is strange
- It is strange that Kurisu called herself Christmas.
- It is strange that Kurisu called herself Mozart.

Figure 1: Clusters of comments given to person-specific errors

Level	Attribute	Relation
Self	(P1) Self-recognition error (Cluster 3)	(P2) Self-relation error (Cluster 4)
Other	(P3) Other-recognition error (Cluster 1)	(P4) Other-relation error (Cluster 2)

Table 3: Matrix of person-specific errors

Figure 1 shows the clustering results of the comments. The figure shows four clusters with the comments that were nearest to the centroid of each cluster, representing salient comments. In Cluster 1, the chatbot was not able to properly discuss topics related to the environment around Kurisu. In Cluster 2, the chatbot suddenly called the user by a different name (e.g., the name of the Kurisu's friend), or gave a name that was irrelevant to the current topic. In Cluster 3, the chatbot provided incorrect information about Kurisu. In Cluster 4, the chatbot mistakenly called Kurisu by a different name. From such an observation, we can see that Clusters1 and 3 are about errors related to the knowledge of the target person, and Clusters 2 and 4 are about errors regarding the misrecognition of relationships. In addition, Clusters 3 and 4 are about errors regarding the target person, while Clusters 1 and 2 are errors about the target person's environment.

On the basis of the above interpretation, we conclude that person-specific errors can be divided into two types: errors in **attributes** (regarding people and things) and errors in **relations** between them, each of which can be further divided into two levels: **self** (the target person) and **other** (surrounding environment of the person such as friends). The matrix of person-specific errors is given in Table 3: (P1) Self-recognition error corresponds to Cluster 3, (P2) Self-relation error corresponds to Cluster 4, (P3) Other-recognition error corresponds to Cluster 1, and (P4) Other-relation error corresponds to Cluster 2.

4.2 Definition of person-specific errors

Each error type is described in detail using actual dialogue examples taken from user interactions with the chatbot. The dialogues were originally in Japanese and translated by the authors. User and the chatbot is denoted as 'U' and the chatbot is denoted as 'S' in the following dialogues.

(P1): Self-recognition error: The error is related to the target person's, i.e., Kurisu's, information (profile, likes/dislikes, experience, etc.).

(1) U: I like you Kurisu.
S: I am neither AI nor AI Kurisu.

(2) U: You are so blunt to me Christina!
S: Hello, I'm AI Christina.

Underlines indicate phrases that are regarded as the cause of the dialogue breakdown by the authors. In dialogue (1), since Kurisu is an AI character, the profile of Kurisu is wrong. In dialogue (2), the chatbot should not have used the nickname "Christina," because Kurisu dislikes it. Her likes and dislikes are wrong.

(P2): Self-relation error: This error means that the chatbot mistook the target person for another person (acquaintance, user, or famous person) or thing.

(3) S: My name is Kurisu Makise. Today's experiment is almost over.
U: I want to talk a lot more.
S: Are you really Kurisu Makise?

(4) U: Who is Mayuri? (Kurisu's friend).
S: She is also known as Kurisu Makise.

In dialogue (3), since the chatbot called the user Kurisu Makise, it mistook itself for another person (i.e., the user). In dialogue (4), the user was asking about Mayuri, but the chatbot talked about itself, mistaking itself for someone else (i.e., Mayuri).

(P3): Other-recognition error: This error indicates the inability to respond to topics of the people surrounding the target person (acquaintances, user, or famous people) or things that the target person is knowledgeable about (favorite things, etc.).

(5) S: Even boys can be cute, you know.
U: Oh, oh. For example?
S: Mayuri, Mayuri, (Kurisu's frined), etc.

(6) U: "nurupo" (Internet slang Kurisu knows)
S: My name is Amadeus Kurisu.

In dialogue (5), since Mayuri is not a boy, the knowledge about the person around Kurisu was incorrect. In dialogue (6), since "nurupo" is an Internet slang that Kurisu knows well, the chatbot has to reply with the traditional response ("ga"), but it was not able to do that.

(P4): Other-relation error: This error is that the chatbot mistook a person or thing other than the target person (an acquaintance, user, or famous person) for another person or thing.

(7) U: I love Amadeus Kurisu.
S: Yes. I like you too. Mayuri.

(8) U: What's Suzuha (Kurisu's friend) doing?
S: Well, Mayuri's voice will save the world.

In dialogue (7), the chatbot called the user "Mayuri", and the system mistook someone other than Kurisu for someone else (in this case, the target person's friend). In dialogue (8), the chatbot responded to a question about Suzuha with Mayuri., i.e., the person (Suzuha) was mistaken for another person (Mayuri).

4.3 Evaluation of person-specific errors

To evaluate the validity of the types of person-specific errors, we investigated inter-annotator agreement in the annotation of the four types (P1–P4). We applied the methods described in

Person-specific error	(I4) Wrong information	(I5) Ignore question	(I9) Ignore expectation	(I10) Unclear intention	(I13) Self-contradiction	(I15) Repetition	etc.	Total
(P1) Self-recognition error	10.1%	1.2%	7.7%	10.7%	4.8%	4.8%	3.0%	42.3%
(P2) Self-relation error	7.1%	0.0%	1.8%	4.2%	0.6%	0.6%	1.2%	15.5%
(P3) Other-recognition error	3.0%	1.8%	7.1%	5.4%	1.2%	2.4%	1.2%	22.1%
(P4) Other-relation error	5.4%	1.2%	1.2%	6.5%	1.8%	3.6%	0.6%	20.3%
Total	25.6%	4.2%	17.8%	26.8%	8.4%	11.4%	6.0%	100.0%

Table 4: Correspondence between person-specific errors and conventional error taxonomy. Percentages show those from total number of person-specific errors (168).

Section 3 to the data not used in the above analysis, resulting in 50 new person-specific error instances obtained from sampled 3,200 system utterances. When annotating the types of person-specific errors, only an utterance labeled as a dialogue breakdown and its preceding three utterances were given to annotators as a context. Two in-house expert annotators conducted the annotation. The definition of person-specific errors described in Section 4.2 was given to the workers as instruction. As a result, the inter-annotator agreement was 0.46 in Cohen's kappa, which indicates a moderate agreement and suggests the validity of the types of person-specific errors.

4.4 Correspondence with existing error types

Table 4 shows the correspondence between person-specific errors and the conventional error taxonomy. The table was created by merging the results shown in Table 2 and Figure 1. Errors on the self-level appeared most frequently, accounting for about half (42.3% + 15.5% = 57.8%) of the person-specific errors. The fact that there were many errors on the others level suggests that the person's environment, such as friends, was also frequently talked about. Each person-specific error corresponded to multiple error types in the conventional taxonomy; thus, we were able to discover different aspects of errors.

The (P1) Self-recognition error was particularly common in (I10) Unclear intention, that is, meaning uttering an unknown intention, such as suddenly changing what the person calls oneself (e.g. from "I" to a nickname). In addition, (P1) Self-recognition error was a common error in (I4) Wrong information, i.e., uttering incorrect information about oneself. The (P2) Self-relation error was also common, especially in (I4) Wrong information, i.e., an error of confusing oneself with a user or oneself with a friend. The (P2) Self-relation error was the next most common in (I10) Unclear intention, such as suddenly mentioning a close friend in a conversation about oneself. In (P3) Other-recognition error and (P4) Other-relation error, there were system utterances of not being able to respond appropriately to topics about people/things the target person is familar with, e.g., incorrect information about them or confusion between users and friends.

From the results of investigating person-specific errors, it became clear that the most common errors were regarding information about the target person then its surrounding environment. Among the error types in the conventional taxonomy, the (I4) Wrong information appeared frequently, confirming the importance of studies on persona-consistent dialogue. In addition to information about the target person, knowledge about the target person's environment is also considered important. Current dialogue systems often do not explicitly model the relationships between people and things, therefore a model that takes into account the knowledge graphs of relationships would be effective (Ghazvininejad et al., 2018; Dinan et al., 2019).

5 Summary and future work

We analyzed dialogue data between a chatbot that imitates a specific person and users to identify person-specific errors that have not been considered thoroughly before. We found that person-specific errors can be divided into four types: self-recognition error, self-relation error, other-recognition error, and other-relation error, which are useful as a guideline for constructing chatbots that are based on specific people.

Future work includes the application of unlikelihood training (Li et al., 2020) or a classifier to estimate the identity of a speaker (Shuster et al., 2021) for suppressing person-specific errors. We focused on one specific person in this paper; thus, it will also be important to consider the generality of the results.

References

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of Wikipedia: Knowledge-powered conversational agents. In *Proc. of ICLR*, pages 1–18.

Takeshi Fuchi and Shinichiro Takagi. 1998. Japanese morphological analyzer using word co-occurrence - JTAG-. In *Proc. of COLING*, pages 409–413.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Proc. of AAAI*, pages 5110–5117.

Ryuichiro Higashinaka, Masahiro Araki, Hiroshi Tsukahara, and Masahiro Mizukami. 2021. Integrated taxonomy of errors in chat-oriented dialogue systems. In *Proc. of SIGDIAL*, pages 89–98.

Ryuichiro Higashinaka, Masahiro Mizukami, Kotaro Funakoshi, Masahiro Araki, Hiroshi Tsukahara, and Yuka Kobayashi. 2015. Fatal or not? finding errors that lead to dialogue breakdowns in chat-oriented dialogue systems. In *Proc. of EMNLP*, pages 2243–2248.

Ryuichiro Higashinaka, Masahiro Mizukami, Hidetoshi Kawabata, Emi Yamaguchi, Noritake Adachi, and Junji Tomita. 2018. Role play-based question-answering by real users for building chatbots with consistent personalities. In *Proc. of SIGDIAL*, pages 264–272.

Koji Inoue, Divesh Lala, Kenta Yamamoto, Shizuka Nakamura, Katsuya Takanashi, and Tatsuya Kawahara. 2020. An attentive listening system with android ERICA: Comparison of autonomous and woz interactions. In *Proc. of SIGDIAL*, pages 118–127.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. In *Proc. of ACL*, pages 994–1003.

Margaret Li, Stephen Roller, Ilia Kulikov, Sean Welleck, Y-Lan Boureau, Kyunghyun Cho, and Jason Weston. 2020. Don't say that! making inconsistent dialogue unlikely with unlikelihood training. In *Proc. of ACL*, pages 4715–4728.

Koh Mitsuda, Ryuichiro Higashinaka, Hiroaki Sugiyama, Masahiro Mizukami, Tetsuya Kinebuchi, Ryuta Nakamura, Noritake Adachi, and Hidetoshi Kawabata. 2021. Fine-tuning a pretrained transformer-based encoder-decoder model with user-generated question-answer pairs to realize character-like chatbots. In *Proc. of IWSDS*, pages 1–14.

Hannah Rashkin, Maarten Sap, and Emily Allaway Noah A. Smith Yejin Choi. 2018. Event2Mind: Commonsense inference on events, intents, and reactions. In *Proc. of ACL*, pages 463–473.

Stephen Roller, Y-Lan Boureau, Jason Weston, Antoine Bordes, Emily Dinan, Angela Fan, David Gunning, Da Ju, Margaret Li, Spencer Poff, et al. 2020. Open-domain conversational agents: Current progress, open problems, and future directions. *arXiv preprint arXiv:2006.12442*.

Kurt Shuster, Jack Urbanek, Arthur Szlam, and Jason Weston. 2021. Am I me or you? state-of-the-art dialogue models cannot maintain an identity. *arXiv preprint arXiv:2112.05843*.

Haoyu Song, Yan Wang, Wei-Nan Zhang, Xiaojiang Liu, and Ting Liu. 2020. Generate, delete and rewrite: A three-stage framework for improving persona consistency of dialogue generation. *arXiv preprint arXiv:2004.07672*.

Hiroaki Sugiyama, Masahiro Mizukami, Tsunehiro Arimoto, Hiromi Narimatsu, Yuya Chiba, Hideharu Nakajima, and Toyomi Meguro. 2021. Empirical analysis of training strategies of transformer-based Japanese chit-chat systems. *arXiv preprint arXiv:2109.05217*.

David Traum, Andrew Jones, Kia Hays, Heather Maio, Oleg Alexander, Ron Artstein, Paul Debevec, Alesia Gainer, Kallirroi Georgila, Kathleen Haase, et al. 2015. New dimensions in testimony: Digitally preserving a holocaust survivor's interactive storytelling. In *Proc. of ICIDS*, pages 269–281.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proc. of ACL*, pages 2204–2213.

Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. 2020. The design and implementation of XiaoIce, an empathetic social chatbot. *Computational Linguistics*, 46(1):53–93.

Direct parsing to sentiment graphs

David Samuel,[1] Jeremy Barnes,[2] Robin Kurtz,[3] Stephan Oepen,[1]
Lilja Øvrelid[1] and Erik Velldal[1]

[1]University of Oslo, Language Technology Group
[2]University of the Basque Country UPV/EHU, HiTZ Center – Ixa
[3]National Library of Sweden, KBLab

{davisamu, oe, liljao, erikve}@ifi.uio.no
jeremy.barnes@ehu.eus, robin.kurtz@kb.se

Abstract

This paper demonstrates how a graph-based semantic parser can be applied to the task of structured sentiment analysis, directly predicting sentiment graphs from text. We advance the state of the art on 4 out of 5 standard benchmark sets. We release the source code, models and predictions.[1]

1 Introduction

The task of structured sentiment analysis (SSA) is aimed at locating all *opinion tuples* within a sentence, where a single opinion contains a) a polar expression, b) an optional holder, c) an optional sentiment target, and d) a positive, negative or neutral polarity, see Figure 1. While there have been sentiment corpora annotated with this information for decades (Wiebe et al., 2005; Toprak et al., 2010), there have so far been few attempts at modeling the full representation, rather focusing on various subcomponents, such as the polar expressions and targets without explicitly expressing their relations (Peng et al., 2019; Xu et al., 2020) or the polarity (Yang and Cardie, 2013; Katiyar and Cardie, 2016).

Dependency parsing approaches have recently shown promising results for SSA (Barnes et al., 2021; Peng et al., 2021). Here we present a novel sentiment parser which, unlike previous attempts, predicts sentiment graphs directly from text without reliance on heuristic lossy conversions to intermediate dependency representations. The model takes inspiration from successful work in meaning representation parsing, and in particular the permutation-invariant graph-based parser of Samuel and Straka (2020) called PERIN.

Experimenting with several different graph encodings, we evaluate our approach on five datasets from four different languages, and find that it compares favorably to dependency-based models across all datasets; most significantly on the more structurally complex ones – **NoReC** and **MPQA**.

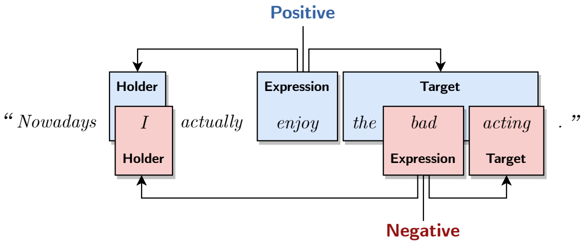


Figure 1: A sentiment graph for the phrase *“Nowadays I actually enjoy the bad acting,”* which contains an example of nesting of two opposing opinions.

2 Related work

Proposing a dependency parsing approach to the full task of SSA, Barnes et al. (2021) show that it leads to strong improvements over state-of-the-art baselines. Peng et al. (2021) propose a sparse fuzzy attention mechanism to deal with the sparseness of dependency arcs in the models from Barnes et al. (2021) and show further improvements. However, in order to apply the parsing algorithm of Dozat and Manning (2018), both of these approaches have to rely on a *lossy* conversion to bi-lexical dependencies with ad-hoc internal head choices for the nodes of the abstract sentiment graph, see Section 3 for a discussion of these issues.

More generally, decoding structured graph information from text has sparked a lot of interest in recent years, especially for parsing meaning representation graphs (Oepen et al., 2020). There has been tremendous progress in developing complex transition-based and graph-based parsers (Hershcovich et al., 2017; McDonald and Pereira, 2006; Dozat and Manning, 2018). In this paper, we adopt PERIN (Samuel and Straka, 2020), a state-of-the-art graph-based parser capable of modeling a superset of graph features needed for our task.

[1] github.com/jerbarnes/direct_parsing_to_sent_graph

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 470 - 478
May 22-27, 2022 ©2022 Association for Computational Linguistics

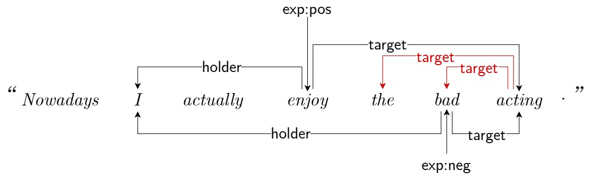


Figure 2: Ambiguous *targets* when encoding the sentence *“Nowadays I actually enjoy the bad acting”* as a head-final bi-lexical dependency graph (Barnes et al., 2021).

3 Issues with dependency encoding

As mentioned above, previous dependency parsing approaches to SSA have relied on a *lossy* bi-lexical conversion. This is caused by an inherent ambiguity in the dependency encoding of two nested text spans with the same head (defined as either the first or the last token in Barnes et al. (2021)).

To be concrete, we can use the running example *“Nowadays I actually enjoy the bad acting,”* which has two opinions with nested targets; *“the bad acting,”*, which is associated with a positive polarity indicated by the polar expression *“enjoy”*, and *“acting,”*, with a negative polarity expressed by *“bad”*. As shown in the dependency representation in Figure 2, both expression–target edges correctly lead to the word *“acting”* but it is impossible to disambiguate the prefix of both targets in the bi-lexical encoding, i.e., to determine that the tokens *“the”* and *“bad”* are part of the target only for the positive opinion. For that, we need a more abstract graph encoding, such as the ones suggested in this paper.

4 PERIN model

PERIN is a general permutation-invariant text-to-graph parser. The output of the parser can be a directed graph with labeled nodes connected by labeled edges where each node is anchored to a span of text (possibly empty or discontinuous). We propose three graph representations for SSA that meet these constrains and thus can be easily modeled by this parser.

We use only a subset of the full PERIN’s functionality for our SSA version – it does not need to use the “relative label rules” and model node properties or edge attributes. Please consult the original work for more technical details about PERIN (Samuel and Straka, 2020).

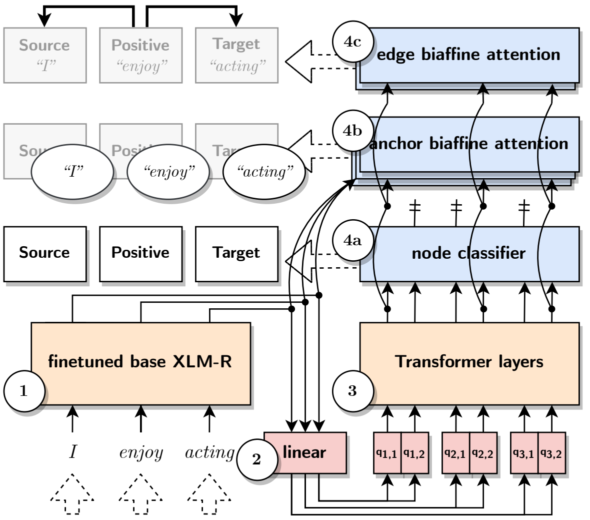


Figure 3: Diagram of the PERIN architecture; 1) each token gets a contextualized embedding and 2) generates queries, 3) queries are further processed and 4) they are put through a) node, b) anchor and c) edge classification heads.

4.1 Architecture

PERIN processes the input text end-to-end in four steps, illustrated in Figure 3: 1) To encode the input, PERIN uses contextualized embeddings from XLM-R (base size; Conneau et al., 2020) and combines them with learned character-level embeddings;[2] 2) each token is mapped onto latent *queries* by a linear transformation; 3) a stack of Transformer (encoder) layers without positional embedding (Vaswani et al., 2017) optionally models the inter-query dependencies; and 4) classification heads select and label queries onto nodes, establish anchoring from nodes to tokens, and predict the node-to-node edges.

4.2 Permutation-invariant query-to-node matching

Traditional graph-based parsers are trained as autoregressive sequence-to-sequence models. PERIN does not assume any prior ordering of the graph nodes.[3] Instead, it processes all queries in parallel and then dynamically maps them to gold nodes.

[2] The character embeddings are not discussed in the PERIN description paper but they are included in the official implementation. We use a single bidirectional GRU layer to process the characters of each token and add the result to the contextualized embeddings. Note that we also excluded them from Figure 3 to simplify the illustration.

[3] Permutation invariance is arguably more important for semantic graphs (with abstract nodes) than for the sentiment graphs. Yet, in case of nested nodes, there is no apparent order, so we do not constrain the model by any ordering assumptions.

Based on the predicted probabilities of labels and anchors, we create a weighted bipartite graph between all queries and nodes. The goal is to find the most probable matching, which can be done efficiently in polynomial time by using the Hungarian algorithm. Finally, every node is assigned to a query and we can backpropagate through standard cross-entropy losses to update the model weights.

4.3 Graph encodings

PERIN defines an overall framework for general graph parsing, it can cater to specific graph encodings by changing the subset of its classification heads. In parsing the abstract sentiment structures, there are several possible lossless graph encodings depending on the positioning of the polarity information and the sentiment node type. We experiment with three variations (Figure 4) and later show that while the graph encoding improves performance, this improvement largely depends on the type of encoding used.

1. **Node-centric encoding**, with labeled nodes and directed unlabeled arcs. Each node corresponds to a target, holder or sentiment expression; edges form their relationships. The parser uses a multi-class node head, an anchor head and a binary edge classification head.
2. **Labeled-edge encoding**, with deduplicated unlabeled nodes and labeled arcs. Each node corresponds to a unique text span from some sentiment graph, while edge labels denote their relationships and functions. The model has a binary node classifier, an anchor classifier and a binary and multi-class edge head.
3. **Opinion-tuple encoding**, which represents the structured sentiment information as a sequence of opinion four-tuples. This encoding is the most restrictive, having the lowest degrees of freedom. The parser utilizes a multi-class node head and three anchor classifiers, it does not need an edge classifier.

5 Data

Following Barnes et al. (2021) we employ five structured sentiment datasets in four languages, the statistics of which are shown in Table 1. The largest dataset is the **NoReC**$_{fine}$ dataset (Øvrelid et al., 2020), a multi-domain dataset of professional reviews in Norwegian. **EU** and **CA** (Barnes et al., 2018) contain hotel reviews in Basque and Catalan, respectively. **MPQA** (Wiebe et al., 2005) annotates news wire text in English. Finally, **DSU** (Toprak et al., 2010) annotates English reviews of online universities. We use the SemEval 2022 releases of **MPQA** and **DSU** (Barnes et al., 2022).[4]

		sentences	**holders**	**targets**	**exps.**	+	**neu**	−
NoReC	train	8634	898	6778	8448	5684	—	2756
	dev	1531	120	1152	1432	988	—	443
	test	1272	110	993	1235	875	—	358
CA	train	1174	169	1695	1981	1272	—	708
	dev	168	15	211	258	151	—	107
	test	336	52	430	518	313	—	204
EU	train	1064	205	1285	1684	1406	—	278
	dev	152	33	153	204	168	—	36
	test	305	58	337	440	375	—	65
MPQA	train	5873	1431	1487	1715	671	337	698
	dev	2063	414	503	581	223	126	216
	test	2112	434	462	518	159	82	223
DSU	train	2253	65	836	836	349	104	383
	dev	232	9	104	104	31	16	57
	test	318	12	142	142	59	12	71

Table 1: Statistics of the datasets, including number of sentences per split, as well as number of holder, target, and polar expression annotations. Additionally, we include the distribution of polarity – restricted to positive, neutral, and negative – in each dataset.

	holders		**targets**		**expressions**	
	#	%	#	%	#	%
NoReC	95	1.5	1187	14.1	1075	9.3
EU	30	2.2	79	4.5	16	0.7
CA	43	2.9	28	1.2	23	0.9
MPQA	48	2.2	250	9.3	145	5.6
DSU	0	0.0	10	1.1	7	0.5

Table 2: Count and percentage of nesting for each dataset.

5.1 Nested dependencies

Returning to the issue of dependency encoding for nested elements discussed in Section 3, Table 2 shows that the amount of nesting in the SSA datasets is not negligible, further motivating our abstract graph encodings for this task.

Table 3a further shows the amount of dependency edges lost because of overlap. Finally, Table 3b shows the SF$_1$ score when converting the gold sentiment graphs to bi-lexical dependency graphs and back – an inherent upper bound for any dependency parser.

[4] Available from `https://competitions.codalab.org/competitions/33556`.

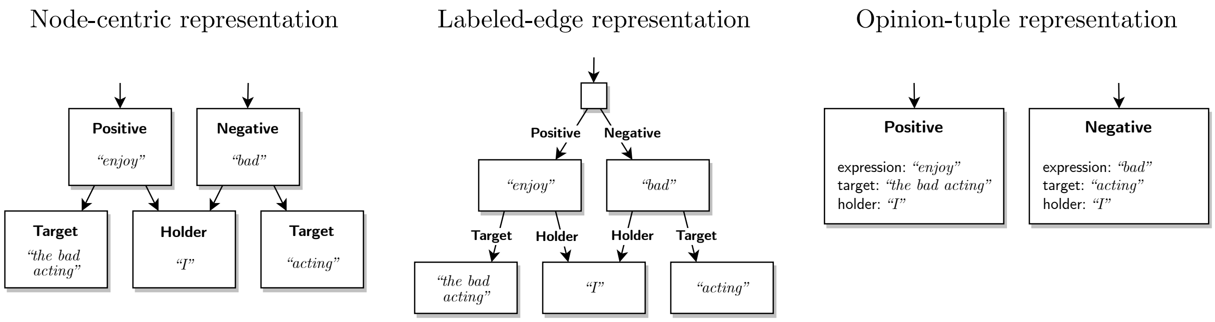


Figure 4: Three representations of the structured sentiment graph for sentence *"Nowadays I actually enjoy the bad acting."*

NoReC	8.8%
EU	4.5%
CA	6.7%
MPQA	4.2%
DSU	0.5%

NoReC	93.6
EU	95.2
CA	97.6
MPQA	96.6
DSU	99.8

Table 3: a) Percentages of dependency arcs lost due to overlap; b) Sentiment Graph F1 after converting test sets to head-final and then reconverting to json format.

6 Experiments

6.1 Evaluation

Following Barnes et al. (2021), we evaluate our models using Sentiment Graph F_1 (**SF**$_1$). This metric considers that each sentiment graph is a tuple of (holder, target, expression, polarity). A true positive is defined as an exact match at graph-level, weighting the overlap in predicted and gold spans for each element, averaged across all three spans. For precision it weights the number of correctly predicted tokens divided by the total number of predicted tokens (for recall, it divides instead by the number of gold tokens). SF_1 allows for empty holders and targets.

In order to further analyze the models, we also include token-level F_1 for extraction of Holders, Targets, and Polar Expressions, as well as Non-polar Sentiment Graph F_1 (**NSF**$_1$).

6.2 Models

We compare our models to the head-final dependency graph parsers from Barnes et al. (2021) as well as the second-order Sparse Fuzzy Attention parser of Peng et al. (2021). For all models, we perform 5 runs with 5 different random seeds and report the mean and standard deviation. Results on development splits are provided in Appendix C, training details are in Appendix D.

6.3 Results

Table 4 shows the main results. Our models outperform both dependency graph models on SF_1, although the results are mixed for span extraction. The opinion-tuple encoding gives the best performance on SF_1 (an average of 6.2 percentage points (pp.) better than Peng et al. (2021)), followed by the labeled edge encoding (3.0) and finally the node-centric encoding (2.1).

For extracting spans, the opinion tuple encoding also achieves the best results on **NoReC**, either labeled-edge or node centric on **CA** and **MPQA**, while Peng et al. (2021) is best on **EU** and **DSU**. This suggests that the main benefit of PERIN is at the structural level, rather than local extraction.

7 Analysis

There are a number of architectural differences between the dependency parsing approaches compared above. In this section, we aim to isolate the effect of predicting intermediate dependency graphs vs. directly predicting sentiment graphs by creating more comparable dependency[5] and PERIN models. We adapt the dependency model from Barnes et al. (2021) by removing the token, lemma, and POS embeddings and replacing mBERT (Devlin et al., 2019) with XLM-R (Conneau et al., 2020). The 'XLM-R dependency' model thus has character LSTM embeddings and token-level XLM-R features. Since these are not updated during training, for the opinion-tuple 'Frozen PERIN' model, we fix the XLM-R weights to make it comparable.

As shown in Table 5, predicting the sentiment graph directly leads to an average gain of 3.7 pp. on the Sentiment Graph F_1 metric. For extracting the

[5] We do not use the model from Peng et al. (2021) as the code is not available.

Dataset	Model	Span F_1 Holder	Span F_1 Target	Span F_1 Exp.	Sent. graph NSF_1 ↑	Sent. graph SF_1 ↑
NoReC	Barnes et al. (2021)	60.4	54.8	55.5	39.2	31.2
	Peng et al. (2021)	63.6	55.3	56.1	40.4	31.9
	PERIN – node-centric	$60.3^{\pm1.8}$	$51.8^{\pm2.5}$	$54.2^{\pm0.9}$	$42.7^{\pm0.6}$	$39.3^{\pm0.7}$
	PERIN – labeled edge	$64.0^{\pm1.5}$	$52.3^{\pm4.2}$	$56.1^{\pm2.7}$	$43.7^{\pm2.2}$	$40.4^{\pm2.1}$
	PERIN – opinion-tuple	$65.1^{\pm2.5}$	*$58.3^{\pm1.5}$	*$60.7^{\pm1.1}$	$47.8^{\pm1.2}$	**41.6**$^{\pm0.7}$
EU	Barnes et al. (2021)	60.5	64.0	72.1	58.0	54.7
	Peng et al. (2021)	65.8	71.0	76.7	66.1	**62.7**
	PERIN – node-centric	$58.9^{\pm1.1}$	$63.5^{\pm1.5}$	$73.9^{\pm0.6}$	$59.8^{\pm0.7}$	$58.6^{\pm0.7}$
	PERIN – labeled edge	$57.6^{\pm2.5}$	$64.9^{\pm0.8}$	$72.5^{\pm1.9}$	$60.0^{\pm1.4}$	$58.8^{\pm1.3}$
	PERIN – opinion-tuple	$64.2^{\pm2.5}$	$67.4^{\pm0.8}$	$73.2^{\pm1.2}$	$62.5^{\pm1.2}$	$61.3^{\pm1.0}$
CA	Barnes et al. (2021)	37.1	71.2	67.1	59.7	53.7
	Peng et al. (2021)	46.2	74.2	71.0	64.5	59.3
	PERIN – node-centric	$56.1^{\pm3.0}$	$69.8^{\pm0.4}$	$70.5^{\pm0.5}$	$63.5^{\pm0.6}$	$61.7^{\pm0.6}$
	PERIN – labeled edge	$60.8^{\pm5.1}$	$70.8^{\pm1.9}$	$72.5^{\pm0.8}$	$64.5^{\pm1.4}$	$62.1^{\pm1.3}$
	PERIN – opinion-tuple	$48.0^{\pm3.9}$	$72.5^{\pm0.7}$	$68.9^{\pm0.2}$	$65.7^{\pm0.7}$	**63.3**$^{\pm0.6}$
MPQA	Barnes et al. (2021)	46.3	49.5	46.0	26.1	18.8
	Peng et al. (2021)	47.9	50.7	47.8	38.6	19.1
	PERIN – node-centric	$58.4^{\pm2.3}$	$60.3^{\pm2.0}$	$55.8^{\pm1.5}$	$38.7^{\pm1.6}$	$28.3^{\pm0.9}$
	PERIN – labeled edge	$53.6^{\pm1.2}$	$53.4^{\pm1.9}$	$53.4^{\pm1.1}$	$33.8^{\pm1.5}$	$27.0^{\pm0.9}$
	PERIN – opinion-tuple	$55.7^{\pm1.7}$	*$64.0^{\pm0.6}$	$53.5^{\pm1.2}$	*$45.1^{\pm1.1}$	***34.1**$^{\pm1.1}$
DSU	Barnes et al. (2021)	37.4	42.1	45.5	34.3	26.5
	Peng et al. (2021)	50.0	44.8	43.7	35.0	27.4
	PERIN – node-centric	$31.4^{\pm5.6}$	$35.0^{\pm1.6}$	$35.1^{\pm2.2}$	$24.8^{\pm0.7}$	$22.9^{\pm1.5}$
	PERIN – labeled edge	$32.5^{\pm6.8}$	$38.0^{\pm3.7}$	$36.2^{\pm2.5}$	$28.8^{\pm2.0}$	$27.3^{\pm1.5}$
	PERIN – opinion-tuple	$42.2^{\pm4.6}$	$40.6^{\pm2.7}$	$39.3^{\pm2.5}$	$33.2^{\pm2.4}$	**31.2**$^{\pm2.4}$

Table 4: Experiments comparing the PERIN model with previous results. We show the average values and their standard deviations from 5 runs. **Bold** numbers indicate the best result for the main SF_1 metric in each dataset. * marks significant difference between our two best approaches, determined by bootstrap testing (see Appendix B).

Dataset	Model	Span F_1 H.	Span F_1 T.	Span F_1 E.	Sent. graph NSF_1	Sent. graph SF_1 ↑
NoReC	XLM-R dependency	58.5	49.9	58.5	37.4	31.9
	Frozen PERIN	48.3	51.9	57.9	*41.8	***35.7**$^{\pm0.6}$
EU	XLM-R dependency	50.0	60.3	70.0	55.1	51.0
	Frozen PERIN	55.5	58.5	68.8	53.1	**51.3**$^{\pm1.2}$
CA	XLM-R dependency	24.9	67.7	67.3	54.8	50.5
	Frozen PERIN	*39.8	69.2	66.3	*60.2	***57.6**$^{\pm1.2}$
MPQA	XLM-R dependency	49.3	*56.9	47.6	30.5	18.9
	Frozen PERIN	44.0	49.0	46.6	30.7	**23.1**$^{\pm1.0}$
DSU	XLM-R dependency	26.8	33.6	36.4	22.9	18.0
	Frozen PERIN	13.8	37.3	33.2	24.5	**21.3**$^{\pm2.9}$

Table 5: Results from comparable experiments, where the dependency graph model (XLM-R dependency) and frozen PERIN models use the same input and similar number of trainable parameters. * marks significant difference, determined by bootstrap (see Appendix B).

spans of holder, target, and polar expressions, the benefit is less clear. Here, the PERIN model only outperforms the XLM-R dependency model 5 of 15 times, which seems to confirm that its benefit is at the graph level. This is further supported by the fact that the highest gains are found on the datasets with the most nested sentiment expressions and dependency arcs lost due to overlap, which are difficult to encode in bi-lexical graphs.

8 Conclusion

Previous work cast the task of structured sentiment analysis (SSA) as dependency parsing, converting the sentiment graphs into lossy dependency graphs. In contrast, we here present a novel sentiment parser which predicts sentiment graphs directly from text without reliance on lossy dependency representations. We adapted a state-of-the-art meaning representation parser and proposed three candidate graph encodings of the sentiment structures. Our experimental results suggest that our approach has clear performance benefits, advancing the state of the art on four out of five standard SSA benchmarks. Specifically, the most direct opinion-tuple encoding provides the highest performance gains. More detailed analysis of the results shows that the benefits stem from better extraction of global structures, rather than local span prediction. Finally, we believe that various structured prediction problems in NLP can similarly be approached in a uniform manner as parsing into directed graphs.

References

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. Structured sentiment analysis as dependency graph parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.

Jeremy Barnes, Laura Ana Maria Oberländer, Enrica Troiano, Andrey Kutuzov, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, and Erik Velldal. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. An Empirical Investigation of Statistical Significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 995–1005, Jeju Island, Korea. Association for Computational Linguistics.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, pages 794–803.

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

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

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2017. A transition-based directed acyclic graph parser for UCCA. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1127–1138, Vancouver, Canada. Association for Computational Linguistics.

Arzoo Katiyar and Claire Cardie. 2016. Investigating LSTMs for joint extraction of opinion entities and relations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 919–929, Berlin, Germany. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy. Association for Computational Linguistics.

Stephan Oepen, Omri Abend, Lasha Abzianidze, Johan Bos, Jan Hajic, Daniel Hershcovich, Bin Li, Tim O'Gorman, Nianwen Xue, and Daniel Zeman. 2020. MRP 2020: The second shared task on cross-framework and cross-lingual meaning representation parsing. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 1–22, Online. Association for Computational Linguistics.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2019. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis.

Letian Peng, Zuchao Li, and Hai Zhao. 2021. Sparse fuzzy attention for structured sentiment analysis.

David Samuel and Milan Straka. 2020. ÚFAL at MRP 2020: Permutation-invariant semantic parsing in PERIN. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 53–64, Online. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

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

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2339–2349, Online. Association for Computational Linguistics.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1640–1649, Sofia, Bulgaria. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample {bert} fine-tuning. In *International Conference on Learning Representations*.

A Changes to datasets

We found out that the official data published at `https://competitions.codalab.org/competitions/33556` was slightly changed from the data used in previous related work. Specifically the **MPQA** and **DSU** datasets had removed a number of errors resulting from the annotation and from the conversion scripts used to create the sentiment graph representations. We re-run the experiments for the comparable baseline model and show the performance differences in Table 7.

B Bootstrap Significance Testing

In order to see whether the performance differences for the experiments are significant, we do bootstrap significance testing Berg-Kirkpatrick et al. (2012), combining two variations. First, we resample the test sets with replacement from all 5 runs together, $b = 1\,000\,000$ times, setting the threshold at $p = 0.05$. Additionally, we test each pair out of the 5×5 combinations for all runs, resampling the test set with replacement $b = 100\,000$ times, setting the threshold again at $p = 0.5$. When one system is significantly better in 15 out of the 25 comparisons, and additionally significantly better in the first joint test, we finally mark it as significantly better.

C Results on development data

To make any future comparison of our approach easier, we show the development scores of all reported models in Table 6.

D Training details

Generally, we follow the training regime described in the original PERIN paper (Samuel and Straka, 2020). The trainable parameters are updated with the AdamW optimizer (Loshchilov and Hutter, 2019), and their learning rate is linearly warmed-up for the first 10% of the training to improve stability, and then decayed with a cosine schedule. The XLM-R parameters are updated with a lower learning rate and higher weight decay to improve generalization. Similarly to PERIN, we freeze the embedding parameters for increased efficiency and regularization. Following the finding by Zhang et al. (2021), we use small learning rates and finetune for a rather long time to increase the training stability. Unlike the authors of PERIN, we did not find any benefits from a dynamic scaling of loss weights (Chen et al., 2018), so we simply set all loss weights to constant 1.0.

We trained our models on a single Nvidia P100 with 16GB memory, the runtimes are given in Table 6. We made five runs from different seeds for each reported value to better estimate the expected error. The hyperparameter configurations for all runs follow, please consult the released code for more details and context: `github.com/jerbarnes/direct_parsing_to_sent_graph`.

General hyperparameters

```
batch_size = 16
beta_2 = 0.98
char_embedding = True
char_embedding_size = 128
decoder_learning_rate = 6.0e-4
decoder_weight_decay = 1.2e-6
dropout_anchor = 0.4
dropout_edge_label = 0.5
dropout_edge_presence = 0.5
dropout_label = 0.85
dropout_transformer = 0.25
dropout_transformer_attention = 0.1
```

Dataset	Model	Span F$_1$			Sent. graph		Runtime	# Params
		Holder	Target	Exp.	NSF$_1$ ↑	SF$_1$ ↑		
NoReC	PERIN – node-centric	54.9$^{\pm4.3}$	52.7$^{\pm2.0}$	57.4$^{\pm1.5}$	44.8$^{\pm1.8}$	$^{p:\,46.4}_{r:\,36.4}$ 40.8$^{\pm1.5}$	9:52 h	108.9 M
	PERIN – labeled edge	59.4$^{\pm2.8}$	52.0$^{\pm2.3}$	57.5$^{\pm2.7}$	44.4$^{\pm1.7}$	$^{p:\,45.7}_{r:\,37.7}$ 41.1$^{\pm1.5}$	9:58 h	109.5 M
	PERIN – opinion-tuple	59.2$^{\pm1.3}$	59.6$^{\pm1.3}$	61.5$^{\pm1.0}$	49.4$^{\pm1.0}$	$^{p:\,42.5}_{r:\,45.5}$ 43.9$^{\pm0.9}$	9:25 h	108.1 M
	Frozen PERIN – opinion-tuple	50.1$^{\pm2.5}$	53.8$^{\pm1.6}$	59.4$^{\pm1.0}$	44.0$^{\pm0.6}$	$^{p:\,33.6}_{r:\,42.2}$ 37.4$^{\pm0.9}$	0:25 h	23.1 M
EU	PERIN – node-centric	57.1$^{\pm3.1}$	68.7$^{\pm1.5}$	69.9$^{\pm1.0}$	61.1$^{\pm1.1}$	$^{p:\,62.8}_{r:\,56.8}$ 59.7$^{\pm1.3}$	1:02 h	87.6 M
	PERIN – labeled edge	51.2$^{\pm4.7}$	66.1$^{\pm2.1}$	66.0$^{\pm1.0}$	59.4$^{\pm1.2}$	$^{p:\,60.1}_{r:\,55.1}$ 57.4$^{\pm1.2}$	0:57 h	88.2 M
	PERIN – opinion-tuple	57.3$^{\pm3.0}$	65.1$^{\pm2.3}$	68.6$^{\pm0.3}$	59.9$^{\pm1.0}$	$^{p:\,64.5}_{r:\,54.7}$ 59.2$^{\pm0.6}$	1:04 h	86.9 M
	Frozen PERIN – opinion-tuple	57.0$^{\pm10.4}$	61.1$^{\pm3.2}$	65.1$^{\pm3.9}$	55.5$^{\pm2.9}$	$^{p:\,56.3}_{r:\,48.8}$ 52.2$^{\pm3.2}$	0:06 h	0.7 M
CA	PERIN – node-centric	57.1$^{\pm2.0}$	73.8$^{\pm2.5}$	74.2$^{\pm1.6}$	68.4$^{\pm2.6}$	$^{p:\,69.9}_{r:\,62.9}$ 66.2$^{\pm2.1}$	1:17 h	87.6 M
	PERIN – labeled edge	48.9$^{\pm4.3}$	72.1$^{\pm0.9}$	72.6$^{\pm1.1}$	67.1$^{\pm1.6}$	$^{p:\,69.5}_{r:\,61.8}$ 65.4$^{\pm1.6}$	1:13 h	88.2 M
	PERIN – opinion-tuple	46.1$^{\pm3.0}$	74.4$^{\pm1.0}$	72.9$^{\pm0.5}$	68.4$^{\pm1.5}$	$^{p:\,73.6}_{r:\,61.6}$ 67.0$^{\pm1.2}$	1:20 h	86.9 M
	Frozen PERIN – opinion-tuple	48.1$^{\pm6.4}$	65.5$^{\pm1.8}$	69.2$^{\pm5.5}$	62.2$^{\pm2.7}$	$^{p:\,64.7}_{r:\,56.0}$ 59.9$^{\pm2.5}$	0:07 h	0.7 M
MPQA	PERIN – node-centric	58.2$^{\pm1.3}$	60.8$^{\pm0.9}$	56.8$^{\pm1.1}$	35.3$^{\pm1.3}$	$^{p:\,34.5}_{r:\,28.7}$ 31.4$^{\pm1.4}$	6:46 h	107.7 M
	PERIN – labeled edge	57.1$^{\pm2.0}$	54.8$^{\pm1.6}$	55.2$^{\pm1.1}$	33.1$^{\pm0.4}$	$^{p:\,35.7}_{r:\,26.4}$ 30.3$^{\pm0.5}$	7:16 h	109.6 M
	PERIN – opinion-tuple	56.0$^{\pm0.6}$	64.2$^{\pm1.7}$	51.7$^{\pm2.8}$	42.1$^{\pm0.8}$	$^{p:\,44.3}_{r:\,30.1}$ 35.8$^{\pm0.6}$	6:43 h	108.1 M
	Frozen PERIN – opinion-tuple	42.0$^{\pm3.8}$	48.1$^{\pm1.7}$	46.6$^{\pm2.6}$	28.1$^{\pm2.2}$	$^{p:\,24.3}_{r:\,20.8}$ 22.2$^{\pm1.5}$	0:37 h	23.1 M
DSU	PERIN – node-centric	0.0$^{\pm0.0}$	41.5$^{\pm4.3}$	40.3$^{\pm2.6}$	27.2$^{\pm2.0}$	$^{p:\,33.4}_{r:\,16.9}$ 22.4$^{\pm1.3}$	2:31 h	107.7 M
	PERIN – labeled edge	0.0$^{\pm0.0}$	46.5$^{\pm1.8}$	41.9$^{\pm3.4}$	28.4$^{\pm2.7}$	$^{p:\,33.2}_{r:\,17.8}$ 23.1$^{\pm2.0}$	2:37 h	109.6 M
	PERIN – opinion-tuple	12.0$^{\pm11.0}$	50.9$^{\pm4.7}$	42.6$^{\pm3.9}$	34.9$^{\pm4.1}$	$^{p:\,39.5}_{r:\,22.6}$ 28.6$^{\pm3.5}$	2:30 h	108.1 M
	Frozen PERIN – opinion-tuple	0.0$^{\pm0.0}$	42.7$^{\pm4.8}$	35.9$^{\pm3.3}$	26.0$^{\pm3.3}$	$^{p:\,29.1}_{r:\,16.3}$ 20.3$^{\pm2.0}$	0:22 h	23.1 M

Table 6: Development scores of all our models from the main section of this paper. SF$_1$ scores are extended by the average precision and recall values. We also show the runtime of a single model and the number of trainable parameters.

Dataset		Span F$_1$			Sent. graph	
		H.	T.	E.	NSF$_1$	SF$_1$
MPQA	original	44.7	51.3	45.7	25.4	15.0
	new data	49.3	56.9	47.6	30.5	18.9
	Δ	+4.6	+5.6	+1.9	+5.1	+4.9
DSU	original	21.0	22.6	35.2	24.0	21.0
	new data	26.8	33.6	36.4	22.9	18.0
	Δ	+5.8	+11.0	+1.3	−1.1	−3.0

Table 7: Results comparing the XLM-R dependency model on the original **MPQA** and **DSU** data, and the new data.

```
dropout_word = 0.1
encoder = "xlm-roberta-base"
encoder_freeze_embedding = True
encoder_learning_rate = 6.0e-6
encoder_weight_decay = 0.1
epochs = 200
focal = True
freeze_bert = False
hidden_size_ff = 4 * 768
hidden_size_anchor = 256
hidden_size_edge_label = 256
hidden_size_edge_presence = 256
layerwise_lr_decay = 0.9
n_attention_heads = 8
n_layers = 3
query_length = 1
pre_norm = True
```

NoReC node-centric hyperparameters

```
graph_mode = "node-centric"
query_length = 2
```

NoReC labeled-edge hyperparameters

```
graph_mode = "labeled-edge"
query_length = 2
```

NoReC opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
```

NoReC frozen opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
freeze_bert = True
batch_size = 8
decoder_learning_rate = 1.0e-4
dropout_transformer = 0.5
epochs = 50
```

EU node-centric hyperparameters

```
graph_mode = "node-centric"
query_length = 2
n_layers = 0
```

EU labeled-edge hyperparameters

```
graph_mode = "labeled-edge"
query_length = 2
n_layers = 0
```

EU opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
n_layers = 0
```

EU frozen opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
freeze_bert = True
n_layers = 0
epochs = 50
```

CA node-centric hyperparameters

```
graph_mode = "node-centric"
query_length = 2
n_layers = 0
```

CA labeled-edge hyperparameters

```
graph_mode = "labeled-edge"
query_length = 2
n_layers = 0
```

CA opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
n_layers = 0
```

CA frozen opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
freeze_bert = True
n_layers = 0
epochs = 50
```

MPQA node-centric hyperparameters

```
graph_mode = "node-centric"
decoder_learning_rate = 1.0e-4
query_length = 2
```

MPQA labeled-edge hyperparameters

```
graph_mode = "labeled-edge"
decoder_learning_rate = 1.0e-4
query_length = 2
```

MPQA opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
```

MPQA frozen opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
freeze_bert = True
batch_size = 8
decoder_learning_rate = 1.0e-4
dropout_transformer = 0.5
epochs = 50
```

DSU node-centric hyperparameters

```
graph_mode = "node-centric"
decoder_learning_rate = 1.0e-4
query_length = 2
```

DSU labeled-edge hyperparameters

```
graph_mode = "labeled-edge"
decoder_learning_rate = 1.0e-4
query_length = 2
```

DSU opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
```

DSU frozen opinion-tuple hyperparameters

```
graph_mode = "opinion-tuple"
freeze_bert = True
batch_size = 8
decoder_learning_rate = 1.0e-4
dropout_transformer = 0.5
epochs = 50
```

XDBERT: Distilling Visual Information to BERT from Cross-Modal Systems to Improve Language Understanding

Chan-Jan Hsu[1,2], **Hung-yi Lee**[1], **Yu Tsao**[2]
[1]National Taiwan University, Taiwan
[2]Academia Sinica, Taiwan
{r09946011, hungyilee}@ntu.edu.tw,
yu.tsao@citi.sinica.edu.tw

Abstract

Transformer-based models are widely used in natural language understanding (NLU) tasks, and multimodal transformers have been effective in visual-language tasks. This study explores distilling visual information from pretrained multimodal transformers to pretrained language encoders. Our framework is inspired by cross-modal encoders' success in visual-language tasks while we alter the learning objective to cater to the language-heavy characteristics of NLU. After training with a small number of extra adapting steps and finetuned, the proposed XDBERT (cross-modal distilled BERT) outperforms pretrained-BERT in general language understanding evaluation (GLUE), situations with adversarial generations (SWAG) benchmarks, and readability benchmarks. We analyze the performance of XDBERT on GLUE to show that the improvement is likely visually grounded.

1 Introduction

Transformer-based models are extensively used in natural language understanding (NLU) tasks, and some prominent pretraining strategies include BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020), and ELECTRA (Clark et al., 2020). Despite their differences in curating the learning objectives, they all utilize text-based datasets only. In the real world, however, humans can benefit from the visual modality when acquiring knowledge from language; an obvious example is learning visually grounded words, such as colors and shapes.

Some studies have succeeded with visually grounded information used in NLU. ViCo (Gupta et al., 2019) learned visual co-occurrences in text and reported superior performance to GloVe in word analogy problems. Zhang et al. (2020) and Huang et al. (2020) used images to boost translation performance in supervised and unsupervised settings. Tan and Bansal (2020) reported improvements over BERT on NLU by proposing the concept of vokenization.

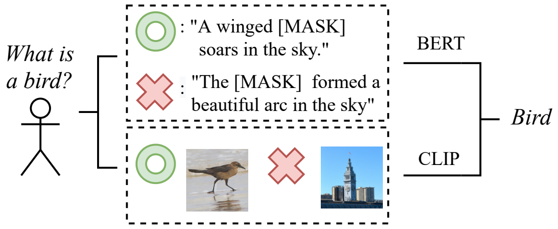

What is a bird?
: "A winged [MASK] soars in the sky."
: "The [MASK] formed a beautiful arc in the sky"
BERT
CLIP
Bird

Figure 1: Humans can answer cloze questions and match a word with an image, and the multi-views of a word could be simulated by neural networks. While BERT excels in masked word reconstruction, CLIP (Section 3) specializes at image-text matching. The two modalities have different collocations of concepts, which incentivize joint learning from the two systems.

Another branch of research focuses on solving multimodal downstream tasks such as visual question answering and image retrieval. Li et al. (2019); Lu et al. (2019); Su et al. (2020); Li et al. (2020) trained visual-text transformers, while LXMERT (Tan and Bansal, 2019) used different encoders for text and image and a cross-modal encoder. Tan and Bansal (2020) tested these models with general language understanding evaluation (GLUE Wang et al. (2018)) and found that the performance does not exceed using BERT (Appendix A), drawing the conclusion that vision-and-language pretraining on visually-grounded language dataset failed to distill useful information for general NLU. CLIP (Radford et al., 2021) utilizes contrastive loss to reach SOTA on zero-shot image classification in a retrieval fashion.

In this work, we establish the link between pretrained multimodal transformers and visually-grounded language learning. We devise a way to distill visual information from components of a pretrained multimodal transformer (CLIP text-transfomer, abbreviated as CLIP-T) to pretrained

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 479 - 489
May 22-27, 2022 ©2022 Association for Computational Linguistics

language transformers (BERT/ELECTRA), to incorporate versatile perception of words into the model (Figure 1). The usage of a visually grounded text-transformer as a teacher allows us to implement straightforward and non-fuzzy adapting tasks for distillation. We show that it is mathematically logical that the CLIP-T output approximates visual features (Sec. 2.2), and also the linguistic competence of CLIP-T is low (Sec. 3), to prove that the distilled information is predominantly visual and thus non-trivial to the pretrained-language transformer despite having textual inputs.

Methodologically, we use the cross-modal encoder structure inspired by Tan and Bansal (2019), to concatenate the two models and further adapt the ensemble for some extra steps (a lot fewer than the original pretraining steps). While adapting pretrained-BERT, we favor a document-level corpus (wiki103) over a vision-language corpus (MSCOCO) due to claims from Devlin et al. (2019)[1] and results from Tan and Bansal (2020) (Appendix A). The adapting tasks are joint masked language modeling (MLM), same sentence prediction, and CLIP token classification tasks, which are resemblant of BERT pretraining tasks to cater to the language-heavy characteristics of NLU. We do ablation studies to show that each of the task provides improvement (Section 5).

During finetuning, we finetune XDBERT (cross-modal distilled BERT), which is the language encoder after adaptation. We evaluate the linguistic capabilities of the model by finetuning on GLUE, situations with adversarial generations (SWAG (Zellers et al., 2018)) benchmarks, and readability benchmarks[2]. The resulting XDBERT outperforms pretrained BERT, proving that our adaptation strategy distills useful visual knowledge into BERT (right of Figure 2). We provide analysis to show that the improvements are visually grounded.

We summarize our contribution as follow:

- We explore distilling visual information from a pretrained multimodal transformer to a pretrained language transformer and improved NLU performance.

- Our adapting method is efficient and extensible to different combinations of pretrained-language encoders (BERT/ELECTRA).

2 Proposed Method

The training process consists of three phases: pretraining, adaptation, and finetuning (Figure 2). Our proposed method focuses on the adaptation phase with pretrained models, so pretraining is not a part of our experiment, but we explain all three phases for completeness. The adaptation phase incorporates the cross-modal transformer structure to jointly learn from CLIP-T and BERT outputs.

2.1 Model Architecture

The cross-modal transformer (middle of Figure 2) consists of a cross-modal encoder, CLIP-T and BERT. CLIP-T has the same module connections as BERT with only parameter differences (specifications in Appendix B). The cross-modal encoder consists of repeating cross-modal encoder layers, which is an extension to single-modality encoder layers (layers of BERT/CLIP-T) in Figure 3. The added cross-attention module follows the attention formula (Vaswani et al., 2017):

$$Attention\ output = softmax\left(\mathbf{Q} * \mathbf{K}^T/\sqrt{D}\right)\mathbf{V} \tag{1}$$

for queries ($\mathbf{Q}$), keys ($\mathbf{K}$) and values ($\mathbf{V}$) of dimension D, however, $\mathbf{Q}$ is generated from a modality other than $\mathbf{K}$ and $\mathbf{V}$. We choose the number of cross-modal encoder layers to be 2.

2.2 Pretraining

BERT is trained using the next sentence prediction and masked language modeling. CLIP is an image-text matching system with two components, a text encoder (CLIP-T), and an image encoder (CLIP-ViT), which learn to encode paired inputs to closer output embeddings via contrastive loss. The trained representation has the following properties:

$$cos(H_i, V_i) >> cos(H_i, V_j)(i \neq j) \tag{2}$$

$$cos(H_i, V_i) >> cos(H_j, V_i)(i \neq j) \tag{3}$$

where H_i is the CLIP text encoder output of X_i, and V_i is the CLIP image encoder output of Y_i. The text-image input (X_i, Y_i) is paired, and every (X_j, Y_k) $(j \neq k)$ is a non-pair. Since H_i and V_i are normalized and have a length of 1, H_i can be used to approximate V_i. The similarity of H_i and V_i is also shown in multi-modal arithmetic propreties discovered in Tewel et al. (2021) Therefore, we use the CLIP text encoder output to approximate CLIP image encoder output for a straightforward adaptation process.

[1] "It is critical to use a document-level corpus rather than a shuffled sentence-level corpus such as the BillionWord Benchmark in order to extract long contiguous sequences"

[2] https://www.kaggle.com/c/commonlitreadabilityprize

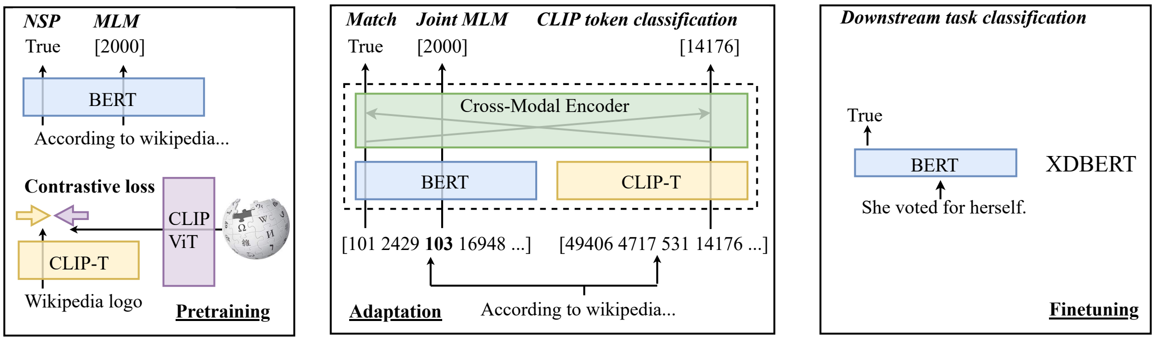


Figure 2: In our experimental setting, the transformers go through three phases of the training processes from left to right. The pretraining phase pretrains BERT and CLIP-T, both of which are then used in the adaptation phase and concatenated with a cross-modal encoder. Finetuning is performed on the language encoder only (XDBERT); in this case, a positive CoLA example is being processed to determine its linguistic acceptability. ViT stands for Vision Transformer (Dosovitskiy et al., 2021), and the input id 103 is the [MASK] token in BERT.

2.3 Adaptation

We define three adapting tasks that can be learned in a self-supervised manner, which is visualized in Figure 2. In these tasks, BERT and CLIP-T takes sentences A and B respectively as input, and losses are calculated from both BERT output and CLIP-T output. Our adapting tasks closely follow BERT text pretraining strategies to retain linguistic competence. Unlike pretraining, the adaptation is computationally inexpensive, as we found that training 1 epoch on wiki103 was already effective. Further training details can be found in Appendix C.

2.3.1 Joint Masked Language Modeling (MLM)

The MLM objective teaches the model to reconstruct masked tokens. The masked ratio and masked token replacement probabilities follow Devlin et al. (2019). Since there is no equivalent of a [MASK] token in CLIP, we leave the sentence as is.

2.3.2 Same sentence prediction (MATCH)

The Image-Text Matching (ITM) objective is widely used in multimodal learning (Tan and Bansal, 2020; Radford et al., 2021). We modify this objective to same sentence prediction as both streams of our model takes text as input. When choosing the input sentences for BERT and CLIP-T, we make the inputs nonidentical 50% of the time. A binary classifier over [CLS] differentiates between the two cases. This motivates the [CLS] output to encode sentence related information, and trains the cross-attention weights.

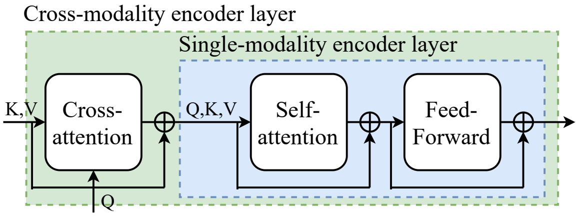


Figure 3: Single-modality encoder layer (blue) and cross-modal encoder layer (green)

2.3.3 CLIP Token Classification

This is the MLM objective done on the CLIP-T side of the full model, omitting the masking part because CLIP has no mask token. Same as MLM, 15% of the tokens are randomly selected for reconstruction. We address concerns on trivial solutions learned by the model in Section 5 and 9 in the appendix.

2.4 Finetuning

Finetuning follows the methods described in Devlin et al. (2019), and is applied to the language encoder only (XDBERT), therefore the number of parameters are kept equal to pretrained-BERT.

3 Experimental Results

We evaluated our model on three NLU benchmarks, namely GLUE, SWAG and READ. We tested our adaptation strategy on three different language encoders coupled with CLIP-T, including BERT-base, ELECTRA-base, and ELECTRA-large. We fix the finetuning parameters between models where comparison is intended, and select the median result of

	RTE	MPRC	STSB	CoLA	SST2	QNLI	QQP	MNLI	SWAG	READ↓
CLIP-T	51.62	76.20	22.07	25.41	–	–	–	–	–	–
BERT-b	66.43	87.38	88.64	56.52	92.46	90.92	89.51	84.35	81.0	–
XDBERT-b	**69.31**	**88.02**	**89.32**	**57.55**	**92.78**	**91.52**	**89.57**	**84.75**	**81.35**	–
ELECTRA-b	78.70	89.49	90.77	66.09	94.5	92.69	90.29	88.23	88.60	–
XDELECTRA-b	**80.51**	**90.55**	**91.04**	**66.76**	**95.20**	**93.03**	**90.4**	**88.75**	**88.73**	–
ELECTRA-l	86.64	91.53	91.88	69.27	96.90	94.78	**91.34**	90.99	92.46	0.685
XDELECTRA-l	**87.73**	**92.12**	**91.97**	**70.98**	**97.36**	**94.93**	91.29	**91.02**	**92.59**	**0.635**

Table 1: NLU task results on the test set (READ) and the dev set (GLUE,SWAG). The results are the median value of 5 runs using different random seeds (9 runs on RTE). BERT-b is the BERT-base-uncased model from Devlin et al. (2019), while XDBERT-b is the proposed models shown in the right part of Figure 2. ELECTRA-b and ELECTRA-l refer to the ELECTRA-base model and the ELECTRA-large model from Clark et al. (2020) respectively. READ (readability benchmark) uses RMSE loss as the evaluation metric.

multiple runs. Details of finetuning are provided in Appendix C.

Table 1 shows experimental results. Each of our XD-model constantly outperforms the original encoder (For fair comparison, we train the original encoder with one more epoch of wiki103). We found that performance gains are more significant on smaller datasets (RTE, MRPC, STSB, CoLA), indicating that visual features help increase generalization when the amount of training data is limited. The gains are also significant on the readability benchmark (READ).

We show that the results of finetuning CLIP-T alone on GLUE does not perform well. Since the language capability of the CLIP-T model is weak, the distilled information obtained by XDBERT/XDELECTRA is predominantly visual.

It is also possible to finetune the entire cross-modal transformer after adaptation. The performance further increases but the model has more parameters. The results are in Appendix C.3.

4 Analysis

To justify the use of a cross-modal encoder, we first conducted a pairwise projection weighted canonical correlation analysis (PWCCA) on word embeddings. The PWCCA is a good measure to determine how close the distributions of two vector groups are to each other. The PWCCA results in Table 2 show low scores on both BERT/CLIP and ELECTRA/CLIP before co-training, so the cross-modal encoder is useful in learning from both distributions.

We inspect RTE, MRPC, and CoLA results of 5 runs in detail to show that the improvements are likely from visual information of CLIP-T. Over the 5 runs, XDBERT-b has accumulated +38 more correct classifications than BERT-b, or +2.74%(38/5/277) gain in performance. MPRC and CoLA show +0.3% and +0.9% gains in accuracy respectively, and translates to a larger gain in performance with their original metric (MRPC F1: +0.83%, CoLA Corr: +2.2%). We then separate each of the glue datasets entries into two categories: entries that XDBERT-b improves classification over BERT-b, and entries of the opposite. Entries where both models obtain the same performance are set aside. Analyzing the separated entries as a whole, we discovered that the better-performing entries have a larger visually grounded ratio (Figure 4), as the quartile, median and mean values are generally higher for improved samples. The enhancement of visually grounded token rep-

Systems	PWCCA
BERT/ELECTRA	0.5498
BERT/CLIP	0.4980
ELECTRA/CLIP	0.4645
BERT/RANDOM	0.3569

Table 2: PWCCA results for different combinations of systems. RANDOM denotes embeddings generated from a uniform distribution.

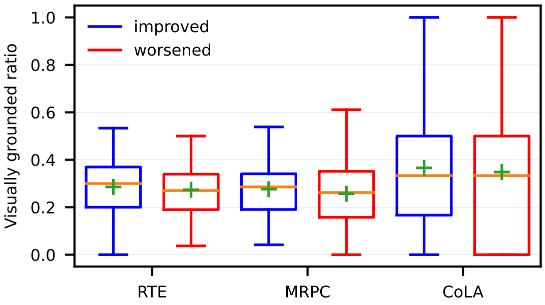


Figure 4: Characteristic analysis of RTE, MRPC, and CoLA entries categorized by performance difference between XDBERT-b and BERT-b. The Green plus symbol denotes the mean value. The visually grounded ratio estimation follows Tan and Bansal (2020).

	RTE	**MPRC**	**STSB**	**CoLA**
MLM+MATCH+CLIPTC(proposed)	69.31	88.02	89.32	56.27
MLM+MATCH	70.04	86.93	88.8	54.62
MLM	68.23	87.25	89.29	54.78
1 cross attention layer	66.79	87.66	89.32	53.62
2 Epochs (2x)	69.31	88.04	89.31	55.91
20 Epochs (20x)	57.4	87.74	-	-
wiki(14G), same steps as above	65.3	87.78	89.1	-

Table 3: Ablation study results. The results are the median value of 5 runs using a learning rate of 1e-4 on XDBERT-b. The CoLA learning rate differs from that in the main paper.

resentations is a rough indicator that XDBERT has obtained distilled visual information from CLIP-T. We show examples of each category in Appendix D.

5 Ablation study

We tried various combinations of adaptation tasks and found out that using all three yielded the best results. We also tried to reduce the number of cross-modal encoder layers to one; however, no further improvements were made upon the visually grounded language encoder. Other experiments include changing the number of layers in the cross-modal encoder, training for longer, and swapping to a much larger wiki (14G). Swapping to wiki reduces potential overfitting from the 20 Epochs setting trained on wiki103, as training for the same amount of steps on wiki is less than 1 epoch. We tested these changes on RTE, MPRC, STSB, and CoLA on 5 random seeds, and the results are shown in Table 3, where MLM refers to the joint MLM objective, MATCH refers to the cross-modal matching objective, and CLIPTC refers to the CLIP token classification objective.

Besides experimental evidence, we also justify the CLIPTC loss via further analysis, as the CLIPTC objective can theoretically be trivially solved by identity mapping. Despite this possibility, we find that the loss is crucial to cross attention learning. Since we do not impose negative hard samples from sampled sentences, the MATCH objective can be solved sufficiently simply by guiding the cross attention to focus on common trivial words. With the CLIPTC objective, the diversity of the input embeddings corresponding to different tokens must be retained in the cross-modal encoder, leading to more robust cross-modal attention. We show comparisons of the attention maps generated from the cross-modal encoders with a random sequence from RTE in Table 9 in the Appendix to verify this claim.

6 Conclusion

In this study, we explored using cross-modal encoders to distill visual information to BERT. We adapted the model with multiple objectives, and we were able to achieve improved performance on NLU tasks. Our adaptation techniques are computationally inexpensive and straightforward. Furthermore, our method is language encoder agnostic, as we show similar performance gains on XDELECTRA.

Acknowledgements

This work was supported in part by Ministry of Science and Technology (MOST), Taiwan, under Contract 110-2223-E-002 -007-MY3.

References

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Tanmay Gupta, Alexander G. Schwing, and Derek Hoiem. 2019. Vico: Word embeddings from visual co-occurrences. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 7424–7433. IEEE.

Po-Yao Huang, Junjie Hu, Xiaojun Chang, and Alexander Hauptmann. 2020. Unsupervised multimodal neural machine translation with pseudo visual pivoting. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8226–8237, Online. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *CoRR*, abs/1908.03557.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pretraining for vision-language tasks. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, pages 121–137. Springer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13–23.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. VL-BERT: pretraining of generic visual-linguistic representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Hao Tan and Mohit Bansal. 2020. Vokenization: Improving language understanding with contextualized, visual-grounded supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2066–2080, Online. Association for Computational Linguistics.

Yoad Tewel, Yoav Shalev, Idan Schwartz, and Lior Wolf. 2021. Zero-shot image-to-text generation for visual-semantic arithmetic. *CoRR*, abs/2111.14447.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326*.

Zhuosheng Zhang, Kehai Chen, Rui Wang, Masao Utiyama, Eiichiro Sumita, Zuchao Li, and Hai Zhao. 2020. Neural machine translation with universal visual representation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

A Visual-Text Transformers Results on NLU

We show the result of Visual-Text Transformers on GLUE, reported by Tan and Bansal (2020) in Table 7. All of the listed methods (except LXMERT) have their text-transformers initialized from BERT. The results show that multi-modal training for solving vision-language tasks does not improve the performance of the models on natural language understanding tasks.

	BERT-b	BERT-l	CLIP
dim	768	1024	512
max_len	512	512	77
#layers	12	24	12

Table 4: BERT and CLIP configurations. ELECTRA has a structure identical to that of BERT. The tokenizers of BERT and CLIP are also different.

B Modeling sequences on CLIP

While BERT and CLIP have similar forwarding mechanisms, the specifications of the transformer architecture are different, resulting in challenges to jointly model both models (Table 4).

Mismatching dimensions pose a problem in cross-attention. We use a linear transformation to generate **Q**, **K**, and **V** of matching dimensions, but clarify that this linear transformation layer exists in the original LXMERT setting where hidden representations have unified dimensions.

We modify the input to address the mismatched max_len of the two systems. In the joint MLM, we used a fixed sequence length of 512 for the BERT. However, the same cannot be done for CLIP as the maxmum model sequence length is 77 for CLIP. We found that most BERT sequences (>99%) of length 512 encode into CLIP sequences of length less than 693, so we pad the CLIP sequence to length 693, and then split the CLIP sequence into 9 sub-sequences of length 77. Therefore, a batch of inputs will contain BERT inputs of size (batch_size, 512) and CLIP inputs of size (batch_size, 9, 77). The output was resized to (batch_size, 693) in the cross-modal encoder. The issue is also present in the finetuning phase, and the maximum sequence length of GLUE and SWAG is 128; therefore we used 2 blocks of CLIP sub-sequences to model it. For bi-sequence classification tasks such as RTE and MRPC, we ensure that separate sentences do not use the same block in the CLIP encoder. Therefore, uni-sequence classification tasks will have a CLIP input size of (batch_size, 2, 77) and the bi-sequence classification task will have a CLIP input size of (batch_size, 4, 77).

C Further Training Details

C.1 Adaptation

We use publicly available wiki103 and preprocessing methods similar to Tan and Bansal (2020) [3]. Wiki103 (500MB) is a subset of the Wikipedia corpus consisting of only good and featured articles. The adaptation of 1 epoch on wiki103 finished in 35 minutes on 8 V100s (BERT-base). We trained for at most 20 epochs(16k steps) and found that further adaptation steps did not increase scores in early epochs, and significantly decreased performance in late epochs. We used the following parameters for adaptation : learning rate = 1e-4, max_epoch = 40 (although we stopped early due to plummeting performance), warmup ratio = 0.05

C.2 Finetuning

The learning rates are listed in Table 5.

	base-sized	large-sized
RTE,MRPC,STSB	1e-4	5e-5
others	2e-5	1e-5

Table 5: Finetuning configurations for NLU tasks. The full model uses the same learning rate as its language encoder

We used a warmup ratio of 0.1, with a learning rate decay of 0.9, and trained the model for 3 epochs. We report the median results of 5 runs on different random seeds, except for RTE, which is unstable; therefore, we report the median results of 9 runs instead. The reproduce results of ELECTRA on RTE and STSB are lower than values reported by Clark et al. (2020) because we did not start from an MNLI checkpoint.

C.3 Finetuning with Full Model

Since our cross-modal transformer itself is can also be viewed as a language encoder, finetuning can be done on the full model. This approach, however, adds extra parameters to pretrained-BERT, so comparison with pretrained-BERT is not intended, instead, we focus on showing the feasibility of this approach. The number of additional parameters is only a function of the hidden size in BERT/ELECTRA, so when the language encoder is large, the ratio of additional parameters is much more insignificant. To simplify notations, we use X-(language encoder) to represent the full model. The

[3] https://github.com/airsplay/vokenization

number of parameters of the full model is shown in Table 6 and the results on NLU tasks are shown in Table 8.

model	parameters
BERT-b / ELECTRA-b	109482240
XBERT-b / XELECTRA-b	202059009
ELECTRA-l	334092288
XELECTRA-l	442671617

Table 6: Number of paramters for each model.

D RTE Examples

We provide three RTE example of each type in Figure4, and we choose extreme examples where performance difference is huge over 5 runs for both "Improved" and "Worsened" categories. We follow Tan and Bansal (2020) to classify tokens as visually-grounded if it is not a stopword and has more than 100 occurrences in MSCOCO. In the following examples, **Bold** words are visually-grounded, while normal words are non-visually-grounded. Words in brackets are stopwords and does not count towards either category.

D.1 Improved : XDBERT outperforms BERT

Example1 :
Visually-grounded ratio : 11/(11+16) = 0.4074
BERT answered correctly : 0/5
XDBERT answered correctly : 5/5

hands across (the) divide (was) formed (in) march 2001 (,) (and) **one** (of) (its) immediate aims (was) (to) press (for) (more) freedom (of) contact (and) communication **right away** (between) (the) **two parts** (of) cyprus (,) (and) (for) early **progress towards** (a) solution (to) (') (the) cyprus problem (') (.)

cyprus (was) divided (into) **two parts** (in) march 2001 (.)

Example2 :
Visually-grounded ratio : 4/(10+4) = 0.2857
BERT answered correctly : 0/5
XDBERT answered correctly : 5/5

(it) (is) hoped (that) **women** (,) (who) constitute (more) (than) **half** (of) (the) population (,) (will) vote (for) (other) **women** (and) ensure (that) (their) issues (are) represented (in) parliament (.)

women (are) poorly represented (in) parliament (.)

Example3 :
Visually-grounded ratio : 13/(13+17) = 0.4333
BERT answered correctly : 0/5
XDBERT answered correctly : 5/5

ho ##dler claimed (there) (were) **also** irregularities (in) (the) campaigns **organized** (by) atlanta (for) (the) 1996 summer **games** (,) sydney (for) (the) summer olympics (in) 2000 (and) salt **lake city** (for) (the) 2002 **winter games** (.)

(before) salt **lake city** (,) **winter** olympic **games** took **place** (in) naga ##no (.)

D.2 On Par : XDBERT and BERT perform equally

Example1 :
Visually-grounded ratio : 6/(6+32) = 0.1375
BERT answered correctly : 0/5
XDBERT answered correctly : 0/5

(on) october 1 2001 (,) eu (and) (other) countries introduced (the) option (for) domestic **animal** owners (to) apply (for) **pet** passports (under) (the) pets **travel** scheme (() pets (for) **short** ()) (,) (for) pets **returning** (from) abroad (to) (the) **united** kingdom (.) (this) replaced (the) **old system**(of) 6 months compulsory qu ##aran ##tine (for) (all) domestic pets (.)

(in) 2001 (,) (the) eu introduced (a) passport(for) pets (.)

Example2 :
Visually-grounded ratio : 5/(5+16) = 0.2381
BERT answered correctly : 5/5
XDBERT answered correctly : 5/5

security forces (were) (on) **high** alert (after) (an) election campaign (in) (which) (more) (than) 1 (,) 000 **people** (,) **including seven** election candidates (,) (have) (been) killed (.)

security forces (were) (on) **high** alert (after) (a) campaign marred (by) violence (.)

Example3 :
Visually-grounded ratio : 8/(8+16) = 0.3333
BERT answered correctly : 5/5
XDBERT answered correctly : 5/5

(in) 1979 (,) (the) leaders signed (the) egypt (-) israel peace treaty (on) (the) **white house lawn** (.) (both) president begin (and) **sad ##at** received (the) nobel peace prize (for) (their)

work (.) (the) **two** nations (have) enjoyed peaceful relations (to) (this) **day** (.)

(the) israel (-) egypt peace agreement (was) signed (in) 1979 (.)

D.3 Worsened : XDBERT underperforms BERT

Example1 :
Visually-grounded ratio : 11/(11+29) = 0.2750
BERT answered correctly : 5/5
XDBERT answered correctly : 0/5

jean (-) claude tri ##chet (,) (the) **european** central **bank** president (,) **made** (it) **clear** (,) (on) wednesday (,) (that) (he) would oppose **un** ##war ##rant **##ed** political **attempts** (to) remove antonio **fa** ##zio (:) (the) **bank** (of) italy governor (,) engulfed (in) controversy (over) (his) handling (of) **bank** takeover bids (.)

antonio **fa** ##zio (is) subordinate (to) jean (-) claude tri ##chet (.)

Example2 :
Visually-grounded ratio : 11/(11+29) = 0.4167
BERT answered correctly : 5/5
XDBERT answered correctly : 0/5

(about) **half** (were) **along** (a) 20 (-) mile stretch (of) **santa** monica **bay** (from) **top** anga canyon boulevard (to) (the) palo **s** verde **s** peninsula (.)

(the) coastline (of) **santa** monica **bay** (is) 50 miles **long** (.)

Example3 :
Visually-grounded ratio : 32/(32+55) = 0.3678
BERT answered correctly : 5/5
XDBERT answered correctly : 0/5

cairo (is) (now) **home** (to) (some) 15 million **people** (-) (a) **bu** ##rgeon **##ing** population (that) produces approximately 10 (,) 000 tonnes (of) rubbish per **day** (,) **putting** (an) enormous strain (on) **public** services (.) (in) (the) **past** 10 years (,) (the) government (has) tried **hard** (to) encourage private investment (in) (the) refuse sector (,) (but) (some) estimate **4** (,) 000 tonnes (of) waste (is) **left behind** every **day** (,) fest ##ering (in) (the) heat (as) (it) **waits** (for) **someone** (to) **clear** (it) (up) (.) (it) (is) often (the) **people** (in) (the) poor ##est neighbourhoods (that) (are) worst affected (.) (but) (in) (some) areas (they) (are) **fighting** **back** (.) (in) shu ##bra (,) **one** (of) (the) northern districts (of) (the) **city** (,) (the) residents (have) **taken** (to) (the) **streets** armed (with) **dust** ##pan **##s** (and) **brushes** (to) **clean** (up) **public** areas (which) (have) (been) **used** (as) **public dump ##s** (.)

15 million tonnes (of) rubbish (are) produced daily (in) cairo (.)

	Diff. to BERT weight	**SST-2**	**QNLI**	**QQP**	**MNLI**
VL-BERT	6.4e-3	90.1	89.5	88.6	82.9
VisualBERT	6.5e-3	90.3	88.9	88.4	82.4
Oscar	41.6e-3	87.3	50.5	86.6	77.3
LXMERT	42.0e-3	82.4	50.5	79.8	31.8
BERT/ViLBERT	–	90.3	89.6	88.4	82.4

Table 7: Results of using Visual-Text Transformers on Natural Language Understanding reported by Tan and Bansal (2020). ViLBERT is identical to BERT because its weights are frozen during multimodal finetuning.

	RTE	**MPRC**	**STSB**	**CoLA**	**SST2**	**QNLI**	**QQP**	**MNLI**	**SWAG**	**READ↓**
XBERT-b	69.31	88.46	89.59	59.05	92.89	91.47	89.37	84.62	81.34	–
XELECTRA-b	79.78	91.06	91.46	66.8	95.06	93.04	90.62	88.97	88.91	–
XELECTRA-l	88.45	92.33	92.04	70.51	97.36	94.97	91.4	91.03	92.83	0.565

Table 8: NLU task results using the full model.

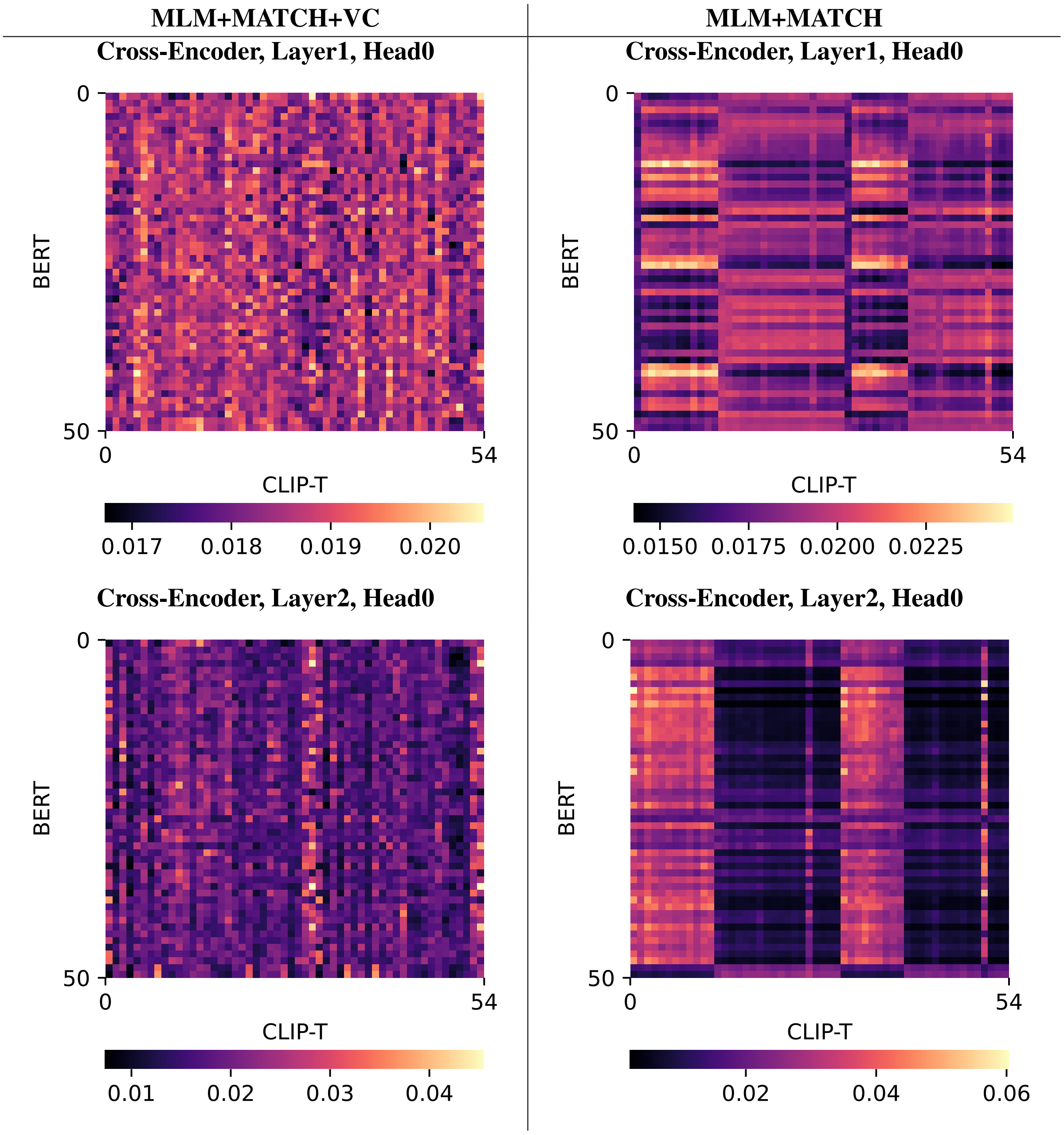


Table 9: Attention map of the cross-attention layers.different experiments. Left: Trained with visual classification loss, Right : Trained without visual classification loss. When trained with VC loss, the different tokens of BERT attends to the different tokens of CLIP-T more diversely.

BERT sequence : [’[CLS]’, ’scientists’, ’had’, ’observed’, ’that’, ’mice’, ’with’, ’a’, ’defective’, ’k’, ’##lot’, ’##ho’, ’gene’, ’aged’,’prematurely’, ’and’, ’wondered’, ’if’, ’an’, ’enhanced’, ’gene’, ’would’, ’have’, ’an’, ’opposite’, ’effect’, ’.’, ’[SEP]’, ’scientists’, ’have’, ’discovered’, ’a’, ’gene’, ’that’, ’produces’, ’a’, ’hormone’, ’that’, ’raises’, ’the’, ’life’, ’expect’, ’##ancy’, ’in’, ’mice’, ’by’, ’30’, ’percent’, ’.’, ’[SEP]’]

CLIP-T sequence : [’<|startoftext|>’, ’scientists’, ’had’, ’observed’, ’that’, ’mice’, ’with’, ’a’, ’defe’, ’ctive’, ’klo’, ’tho’, ’gene’, ’aged’, ’pre’, ’matu’, ’rely’, ’and’, ’wondered’, ’if’, ’an’, ’enhanced’, ’gene’, ’would’, ’have’, ’an’, ’opposite’, ’effect’, ’.’, ’<|endoftext|>’,’<|startoftext|>’, ’scientists’, ’have’, ’discovered’, ’a’, ’gene’, ’that’, ’produces’, ’a’, ’hormone’, ’that’, ’raises’, ’the’, ’life’, ’expect’, ’ancy’, ’in’, ’mice’, ’by’, ’3’, ’0’, ’percent’, ’.’, ’<|endoftext|>’]

As Little as Possible, as Much as Necessary: Detecting Over- and Undertranslations with Contrastive Conditioning

Jannis Vamvas[1] and **Rico Sennrich**[1,2]
[1]Department of Computational Linguistics, University of Zurich
[2]School of Informatics, University of Edinburgh
{vamvas,sennrich}@cl.uzh.ch

Abstract

Omission and addition of content is a typical issue in neural machine translation. We propose a method for detecting such phenomena with off-the-shelf translation models. Using contrastive conditioning, we compare the likelihood of a full sequence under a translation model to the likelihood of its parts, given the corresponding source or target sequence. This allows to pinpoint superfluous words in the translation and untranslated words in the source even in the absence of a reference translation. The accuracy of our method is comparable to a supervised method that requires a custom quality estimation model.

1 Introduction

Neural machine translation (NMT) is susceptible to coverage errors such as the addition of superfluous target words or the omission of important source content. Previous approaches to detecting such errors make use of reference translations (Yang et al., 2018) or employ a separate quality estimation (QE) model trained on synthetic data for a language pair (Tuan et al., 2021; Zhou et al., 2021).

In this paper, we propose a reference-free algorithm based on hypothetical reasoning. Our premise is that a translation has optimal coverage if it uses *as little information as possible and as much information as necessary* to convey the source sequence. Therefore, an addition error means that the source would be better conveyed by a translation containing less information. Conversely, an omission error means that the translation would be more adequate for a less informative source sequence.

Adapting our *contrastive conditioning* approach (Vamvas and Sennrich, 2021), we use probability scores of NMT models to approximate this concept of coverage. We create parse trees for both the source sequence and the translation, and treat their constituents as units of information. Omission errors are detected by systematically deleting constituents from the source and by estimating the probability of the translation conditioned on such a partial source sequence. If the probability score is higher than when the translation is conditioned on the full source, the deleted constituent might have no counterpart in the translation (Figure 1). We apply the same principle to the detection of addition errors by swapping the source and the target sequence.

When comparing the detected errors to human annotations of coverage errors on the segment level (Freitag et al., 2021), our approach surpasses a supervised QE baseline that was trained on a large number of synthetic coverage errors. Human raters find that word-level precision is higher for omissions than additions, with 39% of predicted error spans being precise for English–German translations, and 20% for Chinese–English. False positive predictions can occur especially in cases where the translation has different syntax than the source. We believe our algorithm could be a useful aid whenever humans remain in the loop, for example in a post-editing workflow.

We release the code and data to reproduce our findings, including a large-scale dataset of synthetic coverage errors in English–German and Chinese–English machine translations.[1]

2 Related Work

Coverage errors in NMT Addition and omission of target words have been observed by human evaluation studies in various languages, with omission as the more frequent error type (Castilho et al., 2017; Zheng et al., 2018). They are included as typical translation issues in the Multidimensional Quality Metrics (MQM) framework (Lommel et al., 2014). *Addition* is defined as an accuracy issue where the target text includes text not present in the source, and *omission* is defined as an accuracy

[1] https://github.com/ZurichNLP/coverage-contrastive-conditioning

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 490 - 500
May 22-27, 2022

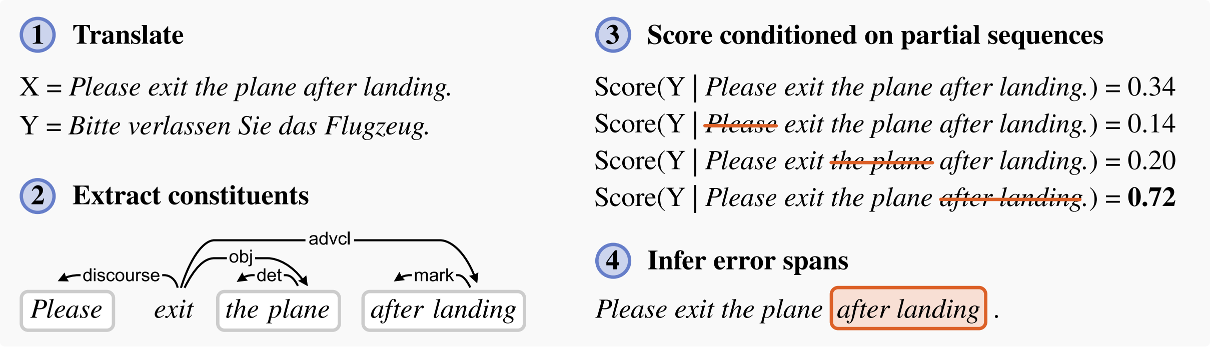

1 Translate
X = *Please exit the plane after landing.*
Y = *Bitte verlassen Sie das Flugzeug.*
2 Extract constituents
advcl
obj
discourse
det
mark
Please
exit
the plane
after landing
3 Score conditioned on partial sequences
Score(Y | *Please exit the plane after landing.*) = 0.34
Score(Y | ~~*Please*~~ *exit the plane after landing.*) = 0.14
Score(Y | *Please exit* ~~*the plane*~~ *after landing.*) = 0.20
Score(Y | *Please exit the plane* ~~*after landing*~~*.*) = **0.72**
4 Infer error spans
Please exit the plane after landing .

Figure 1: Example of how an omission error is detected. German translation Y leaves *after landing* erroneously untranslated (Step 1). Potential error spans are derived from a parse tree (Step 2). An NMT model such as mBART50 assigns a higher probability score to Y conditioned on the source with *after landing* deleted than to Y conditioned on the full source (Step 3). This indicates that there is an omission error (Step 4).

issue where content is missing from the translation but is present in the source.[2]

Freitag et al. (2021) used MQM to manually re-annotate English–German and Chinese–English machine translations submitted to the WMT 2020 news translation task (Barrault et al., 2020). Their findings confirm that state-of-the-art NMT systems still erroneously add and omit target words, and that omission occurs more often than addition. Similar patterns can be found in English–French machine translations that have been annotated with fine-grained MQM labels for the document-level QE shared task (Specia et al., 2018; Fonseca et al., 2019; Specia et al., 2020).

Detecting and reducing coverage errors While reference-based approaches include measuring the n-gram overlap to the reference (Yang et al., 2018) and analyzing word alignment to the source (Kong et al., 2019), this work focuses on the *reference-free* detection of coverage errors.

Previous work has employed custom QE models trained on labeled parallel data. For example, Zhou et al. (2021) insert synthetic hallucinations and train a Transformer to predict the inserted spans. Similarly, Tuan et al. (2021) train a QE model on synthetically noisy translations. In this paper, we propose a method that is based on off-the-shelf NMT models only.

Other related work has focused on improving coverage during decoding or training, for example via attention (Tu et al., 2016; Wu et al., 2016; Li et al., 2018; among others). More recently, Yang et al. (2019) found that contrastive fine-tuning on references with synthetic omissions reduces coverage errors produced by an NMT system.

3 Approach

Contrastive Conditioning Properties of a translation can be inferred by estimating its probability conditioned on contrastive source sequences (Vamvas and Sennrich, 2021). For example, if a certain translation is more probable under an NMT model when conditioned on a counterfactual source sequence, the translation might be inadequate.

Application to Omission Errors Figure 1 illustrates how contrastive conditioning can be directly applied to the detection of omission errors. We construct *partial source sequences* by systematically deleting constituents from the source. If the probability score of the translation (average token log-probability) is higher when conditioned on such a partial source, the deleted constituent is taken to be missing from the translation.

To compute the probability score for a translation Y given a source sequence X, we sum up the log-probabilities for every target token and normalize the sum by the number of target tokens:

$$\text{score}(Y|X) = \frac{1}{|Y|}\sum_{i=0}^{|Y|} \log p_\theta(y_i|X, y_{<i})$$

Application to Addition Errors We apply the same method to addition detection, but swap the source and target languages. Namely, we use an NMT model for the reverse translation direction, and we score the source sequence conditioned on the full translation and a set of partial translations.[3]

[2] The terms *overtranslation* and *undertranslation* have been used in the literature as well. MQM reserves these terms for errors where the translation is too specific or too unspecific.

[3] Another possibility would be to leave the translation direction unreversed and to score the partial translations con-

Potential Error Spans In its most basic form, our algorithm does not require any linguistic resources apart from tokenization. For a source sentence of n tokens one could create n partial source sequences with the ith token deleted. However, such an approach would rely on a radical assumption of compositionality, treating all tokens as independent constituents.

We thus propose to extract potential error spans from parse trees, specifically from dependency trees predicted by Universal Dependency parsers (de Marneffe et al., 2021), which are widely available. This allows (a) to skip function words and (b) to include a reasonable number of multi-word spans in the set of potential error spans. Formally, we consider word spans that satisfy the following conditions:

1. A potential error span is a complete subtree of the dependency tree.
2. It covers a contiguous subsequence.
3. It contains a part of speech of interest.

For every potential error span, we create a partial sequence by deleting the span from the original sequence. This is still a simplified notion of constituency, since some partial sequences will be ungrammatical. Our assumption is that NMT models can produce reliable probability estimates despite the ungrammatical input.

4 Experimental Setup

In this section we describe the data and tools that we use to implement and evaluate our approach.

Scoring model We use mBART50 (Tang et al., 2021), which is a sequence-to-sequence Transformer pre-trained on monolingual corpora in many languages using the BART objective (Lewis et al., 2020; Liu et al., 2020) that was fine-tuned on English-centric multilingual MT in 50 languages. Sequence-level probability scores are computed by averaging the log-probabilities of all target tokens. We use the one-to-many mBART50 model if English is the source language, and the many-to-one model if English is the target language.

Error spans We use Stanza (Qi et al., 2020) for dependency parsing, a neural pipeline for various languages trained on data from Universal Dependencies (de Marneffe et al., 2021). We make use of universal part-of-speech tags (UPOS) to define parts of speech that might constitute potential error spans. Specifically, we treat common nouns, proper nouns, main verbs, adjectives, numerals, adverbs, and interjections as relevant parts of speech.

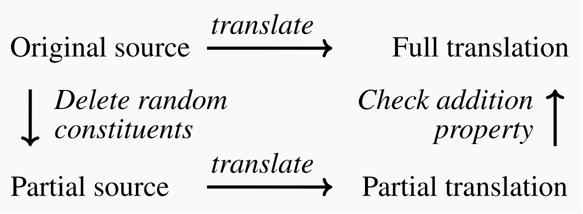


Figure 2: Process designed for creating machine translations with synthetic coverage errors. The full translation contains an addition error with regard to the partial source, and the partial translation contains an omission error with regard to the original source sequence.

Gold Standard Data We use state-of-the-art English–German and Chinese–English machine translations for evaluation, which have been annotated by Freitag et al. (2021) with translation errors.[4] We set aside translations by the system *Online-B* as a development set, and use the other systems as a test set, excluding translations by humans. The development set was used to identify the typical parts-of-speech of coverage error spans, listed in the paragraph above.

Synthetic Data We also create synthetic coverage errors, which we use for training a supervised baseline QE system. We propose a data creation process that is inspired by previous work (Yang et al., 2019; Zhou et al., 2021; Tuan et al., 2021) but is defined such that it works for both additions and omissions, and produces fluent translations.

Figure 2 illustrates the process. We start from the original source sentences and create *partial sources* by deleting randomly selected constituents. Specifically, we delete each constituent with a probability of 15%. We then machine-translate both the original and the partial sources, yielding *full* and *partial machine translations*. We retain only samples where the full machine translation is different from the partial one, and can be constructed by addition.

This allows us to treat the full translations as overtranslations of the partial sources, and the added words as addition errors. Conversely, the partial translations are treated as undertranslations of the original sources. Negative examples are cre-

ditioned on the source. However, the scores might be confounded by a lack of fluency in the partial translations.

[4] https://github.com/google/wmt-mqm-human-evaluation

	Approach	Detection of additions			Detection of omissions		
		Precision	Recall	F1	Precision	Recall	F1
EN–DE	Supervised baseline	6.9±1.9	2.9±0.9	4.0±1.3	40.3±5.2	6.1±0.1	10.6±0.2
	Our approach	4.0	15.0	**6.3**	22.3	18.8	**20.4**
ZH–EN	Supervised baseline	4.3±0.6	4.7±0.7	**4.5**±0.6	49.6±0.6	9.4±1.0	15.9±1.4
	Our approach	1.7	40.6	3.4	25.8	62.0	**36.5**

Table 1: Segment-level comparison of coverage error detection methods on the gold dataset by Freitag et al. (2021). We average over three baseline models trained with different random seeds, reporting the standard deviation.

ated by pairing the original sources with the full translations, and the partial sources with the partial translations.[5]

Our synthetic data are based on monolingual news text released for WMT.[6] To train the baseline system, we use 80k unique source segments per language pair. Statistics are reported in Table A3.

Supervised baseline system Following the approach outlined by Moura et al. (2020), we use the OpenKiwi framework (Kepler et al., 2019) to train a separate Predictor-Estimator model (Kim et al., 2017) per language pair, based on XLM-RoBERTa (Conneau et al., 2020). The supervised task can be described as token-level binary classification. Every token is classified as either `OK` or `BAD`, similar to the word-level labels used for the QE shared tasks (Specia et al., 2020). A source token is `BAD` if it is omitted in the translation, and a token in the translation is `BAD` if it is part of an addition error. For English and German, we use the Moses tokenizer (Koehn et al., 2007) to separate the text into labeled tokens; for Chinese we label the text on the character level.

Where suitable, we use the default settings of OpenKiwi. We fine-tune the large version of XLM-RoBERTa, which results in a model of similar parameter count as the mBART50 model we use for contrastive conditioning. We train for 10 epochs with a batch size of 32, with early stopping on the validation set. For token classification we train two linear layers, separately for source and target language (which corresponds to omissions and additions, respectively). We use AdamW (Loshchilov and Hutter, 2019) with a learning rate of 1e-5, freezing the pretrained encoder for the first 1000 steps.

5 Evaluation

5.1 Segment-Level Comparison to Gold Data

The accuracy of our approach can be estimated based on the human ratings by Freitag et al. (2021).

Evaluation Design We use the MQM error types *Accuracy/Addition* and *Accuracy/Omission*, and ignore other types such as *Accuracy/Mistranslation*. We count a prediction as correct if any one of the human raters has marked the same error type anywhere in the segment.[7] We exclude segments from the evaluation that might have been incompletely annotated (because raters stopped after marking five errors). For ease of implementation, we also exclude segments that consist of multiple sentences.

Results The results of the gold-standard comparison are shown in Table 1. Our approach clearly surpasses the baseline in the detection of omission errors in both language pairs. However, both approaches recognize addition errors with low accuracy, and especially the supervised baseline has low recall. Considering its high performance on a synthetic test set (Table A1 in the Appendix), it seems that the model does not generalize well to real-world coverage errors, highlighting the challenges of training a supervised QE model on purely synthetic data.

5.2 Human Evaluation of Precision

We perform an additional word-level human evaluation to analyze the predictions obtained via our approach in more detail. Our human raters were presented segments that had been marked as true or false positives in the above evaluation, allowing us to quantify word-level precision.

[5] Note that the synthetic dataset does not contain translations with both an addition and an omission error, which is a limitation. Still, we expect that a system trained on the dataset will be able to generalize to such examples, especially if two separate classifiers are used for additions and omissions.

[6] `http://data.statmt.org/news-crawl/`

[7] We perform a segment-level evaluation and do not quantify word-level accuracy in this section since the dataset does not contain consistently annotated spans for coverage errors.

		EN–DE	ZH–EN
Target	Addition errors	2.3	1.2
	Any errors	7.4	12.0
Source	Omission errors	36.3	13.8
	Any errors	39.4	19.5

Table 2: Human evaluation: word-level precision of the spans that were highlighted by our approach.

Evaluation Design We employed two linguistic experts per language pair as raters.[8] Each rater was shown around 700 randomly sampled positive predictions across both types of coverage errors.

Raters were shown the source sequence, the machine translation, and the predicted error span. They were asked whether the highlighted span was indeed translated badly, and were asked to perform a fine-grained analysis based on a list of predefined answer options (Figures 3 and 4 in the Appendix).

A part of the samples were annotated by both raters. The agreement was moderate for the main question, with a Cohen's kappa of 0.54 for English–German and 0.45 for Chinese–English. Agreement on the more subjective follow-up question was lower (0.32 / 0.13).

Results The fine-grained answers allow us to quantify the word-level precision of the spans highlighted by our approach, both with respect to coverage errors in particular and to translation errors in general (Table 2). Precision is higher than expected when detecting omission errors in English–German translations, but is still low for additions. The distribution of the detailed answers (Figures 3 and 4 in the Appendix) suggests that syntactical differences between the source and target language contribute to the false positives regarding additions. Example predictions are provided in Appendix F, which include cases where all three raters of Freitag et al. (2021) had overlooked the coverage error.

Finally, Table 2 shows that many of the predicted error spans are in fact translation errors, but not coverage errors in a narrow sense. For example, more than 10% of the spans marked in Chinese–English translations were classified by our raters as a different type of accuracy error, such as mistranslation.

6 Limitations and Future Work

We hope that the automatic detection of coverage errors could be an aid to translators and post-editors, given that manually detecting such errors is tedious. Our results on omissions are encouraging, and user studies are recommended in order to validate the usefulness of the predictions to practitioners. Further work needs to be done to improve the detection of additions, of which the real-world data contain few examples. Higher accuracy would be necessary for word-level QE to be helpful (Shenoy et al., 2021), and so with regard to detecting addition errors, the practical utility of both the baseline and of our approach remains limited.

Inference time should also be discussed. In Appendix C we perform a comparison, finding that on a long sentence pair contrastive conditioning can take up to ten times longer than a forward pass of the baseline. However, this is still a fraction of the time needed for generating a translation in the first place. In addition, restricting the potential error spans that are considered could further improve efficiency.

7 Conclusion

We have proposed a reference-free method to automatically detect coverage errors in translations. Derived from contrastive conditioning, our method relies on hypothetical reasoning over the likelihood of partial sequences. Since any off-the-shelf NMT model can be used to estimate conditional likelihood, no access to the original translation system or to a quality estimation model is needed. Evaluation on real machine translations shows that our approach outperforms a supervised baseline in the detection of omissions. Future work could address the low precision on addition errors, which are relatively rare in the datasets we used for evaluation.

Acknowledgments

This work was funded by the Swiss National Science Foundation (project MUTAMUR; no. 176727). We would like to thank Xin Sennrich for facilitating the recruitment of annotators, and Chantal Amrhein as well as the anonymous reviewers for helpful feedback.

References

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette

[8] Raters were paid ca. USD 30 per hour.

Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. 2020. Findings of the 2020 conference on machine translation (WMT20). In *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online. Association for Computational Linguistics.

Sheila Castilho, Joss Moorkens, Federico Gaspari, Rico Sennrich, Vilelmini Sosoni, Yota Georgakopoulou, Pintu Lohar, Andy Way, Antonio Miceli Barone, and Maria Gialama. 2017. A comparative quality evaluation of PBSMT and NMT using professional translators. *16th Machine Translation Summit 2017*, pages 116–131.

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

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Erick Fonseca, Lisa Yankovskaya, André F. T. Martins, Mark Fishel, and Christian Federmann. 2019. Findings of the WMT 2019 shared tasks on quality estimation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 1–10, Florence, Italy. Association for Computational Linguistics.

Markus Freitag, George Foster, David Grangier, Viresh Ratnakar, Qijun Tan, and Wolfgang Macherey. 2021. Experts, Errors, and Context: A Large-Scale Study of Human Evaluation for Machine Translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474.

Fabio Kepler, Jonay Trénous, Marcos Treviso, Miguel Vera, and André F. T. Martins. 2019. OpenKiwi: An open source framework for quality estimation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 117–122, Florence, Italy. Association for Computational Linguistics.

Hyun Kim, Jong-Hyeok Lee, and Seung-Hoon Na. 2017. Predictor-estimator using multilevel task learning with stack propagation for neural quality estimation. In *Proceedings of the Second Conference on Machine Translation*, pages 562–568, Copenhagen, Denmark. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Xiang Kong, Zhaopeng Tu, Shuming Shi, Eduard Hovy, and Tong Zhang. 2019. Neural machine translation with adequacy-oriented learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6618–6625.

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

Yanyang Li, Tong Xiao, Yinqiao Li, Qiang Wang, Changming Xu, and Jingbo Zhu. 2018. A simple and effective approach to coverage-aware neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 292–297, Melbourne, Australia. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual Denoising Pre-training for Neural Machine Translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Arle Lommel, Hans Uszkoreit, and Aljoscha Burchardt. 2014. Multidimensional quality metrics (MQM): A framework for declaring and describing translation quality metrics. *Tradumàtica*, (12):0455–463.

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

João Moura, Miguel Vera, Daan van Stigt, Fabio Kepler, and André F. T. Martins. 2020. IST-unbabel participation in the WMT20 quality estimation shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1029–1036, Online. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational*

Linguistics: System Demonstrations, pages 101–108, Online. Association for Computational Linguistics.

Raksha Shenoy, Nico Herbig, Antonio Krüger, and Josef van Genabith. 2021. Investigating the helpfulness of word-level quality estimation for post-editing machine translation output. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10173–10185, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lucia Specia, Frédéric Blain, Marina Fomicheva, Erick Fonseca, Vishrav Chaudhary, Francisco Guzmán, and André F. T. Martins. 2020. Findings of the WMT 2020 shared task on quality estimation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 743–764, Online. Association for Computational Linguistics.

Lucia Specia, Frédéric Blain, Varvara Logacheva, Ramón F. Astudillo, and André F. T. Martins. 2018. Findings of the WMT 2018 shared task on quality estimation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 689–709, Belgium, Brussels. Association for Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2021. Multilingual translation from denoising pre-training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3450–3466, Online. Association for Computational Linguistics.

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

Yi-Lin Tuan, Ahmed El-Kishky, Adithya Renduchintala, Vishrav Chaudhary, Francisco Guzmán, and Lucia Specia. 2021. Quality estimation without human-labeled data. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 619–625, Online. Association for Computational Linguistics.

Jannis Vamvas and Rico Sennrich. 2021. Contrastive conditioning for assessing disambiguation in MT: A case study of distilled bias. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10246–10265, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Jing Yang, Biao Zhang, Yue Qin, Xiangwen Zhang, Qian Lin, and Jinsong Su. 2018. Otem&Utem: Over- and under-translation evaluation metric for NMT. In *Natural Language Processing and Chinese Computing*, pages 291–302, Cham. Springer International Publishing.

Zonghan Yang, Yong Cheng, Yang Liu, and Maosong Sun. 2019. Reducing word omission errors in neural machine translation: A contrastive learning approach. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6191–6196, Florence, Italy. Association for Computational Linguistics.

Zaixiang Zheng, Hao Zhou, Shujian Huang, Lili Mou, Xinyu Dai, Jiajun Chen, and Zhaopeng Tu. 2018. Modeling Past and Future for Neural Machine Translation. *Transactions of the Association for Computational Linguistics*, 6:145–157.

Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Francisco Guzmán, Luke Zettlemoyer, and Marjan Ghazvininejad. 2021. Detecting hallucinated content in conditional neural sequence generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1393–1404, Online. Association for Computational Linguistics.

A Annotator Guidelines

You will be shown a series of source sentences and translations. One or several spans in the text are highlighted and it is claimed that the spans are translated badly. You are asked to determine whether the claim is true. The highlighted spans can be either in the source sequence or in the translation. If a span is in the source sentence, check whether it has been correctly translated. If a span is in the translation, check whether it correctly conveys the source. Sometimes, multiple spans are highlighted. In that case, focus your answer on the span that is most problematic for the translation. In a second step, you are asked to select an explanation. On the one hand, if you agree that the highlighted span is translated badly, please explain your reasoning by selecting your explanation. On the other hand, if you disagree and think that the span is well-translated, please select an explanation why the span might have been marked as badly translated in the first place. Should multiple explanations be equally plausible, select the first from the top.

	Detection of additions				Detection of omissions			
	Prec.	*Recall*	*F1*	*MCC*	*Prec.*	*Recall*	*F1*	*MCC*
EN–DE								
Supervised Baseline	98.8±0.4	98.0±.2	**98.4**±.2	**96.8**±.1	94.0±1.3	96.6±0.4	**95.3**±.5	**90.5**±.2
Ours	78.1	88.3	82.9	76.7	80.9	98.6	88.9	78.1
ZH–EN								
Supervised Baseline	87.2±1.5	75.7±.6	**81.0**±.3	**72.6**±.6	67.3±1.3	68.0±1.2	**67.7**±.9	**53.8**±.3
Ours	26.1	88.9	40.4	23.3	28.3	92.0	43.3	40.3

Table A1: Segment-level and word-level (*MCC*) evaluation based on a test set with synthetic coverage errors.

	Short sentence pair			Long sentence pair		
	Additions	Omissions	Both	Additions	Omissions	Both
Supervised baseline	-	-	25 ms	-	-	25 ms
Our approach	40 ms	45 ms	83 ms	165 ms	197 ms	365 ms
– excluding parser	18 ms	21 ms	38 ms	102 ms	144 ms	239 ms

Table A2: Inference times when predicting on a short and a long sentence pair. Since we did not use a parser that is optimized for efficiency, we additionally report inference time without including the time needed for parsing.

B Evaluation on Synthetic Errors

We used a test split held back from the synthetic data to perform an additional evaluation. On the segment level, we report Precision, Recall and F1-score. Like in Section 5.1, a prediction is treated as correct on the segment level if for a predicted coverage error there is indeed a coverage error of that type anywhere in the segment.

On the word level, we follow previous work on word-level QE (Specia et al., 2020) and report the Matthews correlation coefficient (MCC) across all the tokens in the test set.

Results Results are shown in Table A1. The supervised baseline has a high accuracy on English–German translations and a moderate accuracy on Chinese–English translations. In comparison, our approach performs clearly worse than the supervised baseline on the synthetic errors.

C Inference Time

Inference times are reported in Table A2. We measure the time needed to run the coverage error detection methods on a short sentence pair and on a long sentence pair for English–German. The short sentence pair is taken from Figure 1 and the long sentence pair has 40 tokens in the source sequence and 47 tokens in the target sequence. We average over 1000 repetitions on RTX 2080 Ti GPUs.

The higher inference times for our approach can be explained by the number of translation probabilities that need to be estimated. On average, we compute 30 scores per sentence in the English–German MQM dataset, and 44 per sentence in the Chinese–English MQM dataset. Still, the time needed for computing all these scores is only a fraction of the time it takes to generate a translation (254 ms for the short source sentence and 861 ms for the long sentence, assuming a beam size of 5).

The required number of scores could be reduced by considering fewer potential error spans. Furthermore, scoring could be parallelized across batches of multiple translations. Finally, using a more efficient parser, or no parser at all, could speed up inference.

D Dataset Statistics

Dataset split	Number of segments			Number of tokens			
	Total	W/ addition	W/ omission	Src. `OK`	Src. `BAD`	Tgt. `OK`	Tgt. `BAD`
EN–DE Train	135269	18423	18423	2185918	58378	2197843	53911
EN–DE Dev	16984	2328	2328	273311	7398	275156	6781
EN–DE Test	16984	2328	2328	273277	7701	275036	7032
ZH–EN Train	110195	10697	10697	2576135	62311	1866567	37730
ZH–EN Dev	14149	1383	1383	326743	7562	236685	4244
ZH–EN Test	14026	1342	1342	322000	7566	234757	4882

Table A3: Statistics for the dataset of synthetic coverage errors described in Section 4.

Dataset split	Number of segments		
	Total	With an addition error	With an omission error
EN–DE Dev	1418	77	187
EN–DE Test	8508	407	1057
– without excluded segments	4839	162	484
ZH–EN Dev	1999	69	516
ZH–EN Test	13995	329	3360
– without excluded segments	8851	149	1569

Table A4: Statistics for the gold dataset by Freitag et al. (2021).

E Examples of Synthetic Coverage Errors

English–German Example

Addition error

Partial source: But they haven’t played.
Full machine translation: Aber sie haben nicht gegen ein Team wie uns gespielt.

Omission error

Full source: But they haven’t played against a team like us.
Partial machine translation: Aber sie haben nicht gespielt.

Chinese–English Example

Addition error

Partial source: 医院和企业共同研发相关检测试剂盒，惠及更多患者。
Full translation: Hospitals and enterprises jointly develop related test kits to benefit more cancer patients.

Omission error

Full source: 医院和企业共同研发相关检测试剂盒，惠及更多肿瘤患者。
Partial translation: Hospitals and enterprises jointly develop related test kits to benefit more patients.

F Examples of Coverage Errors Predicted by Contrastive Conditioning

English–German Examples

Predicted addition error

Source: He added: "It’s backfired on him now, though, that’s the sad thing."
Machine translation: Er fügte **hinzu**: "Es ist jetzt auf ihn abgefeuert, aber das ist das Traurige."

Original MQM rating (Freitag et al., 2021): No related accuracy error marked by the three raters.

Answer by our human rater: The highlighted target span is not translated badly. It might have been highlighted because it is syntactically different from the source.

Meaning of highlighted span: hinzu = ‘additionally’

Predicted omission error

Source: UK’s medical **drug** supply still uncertain in no-deal Brexit
Machine translation: Die medizinische Versorgung Großbritanniens ist im No-Deal-Brexit noch ungewiss

Original MQM rating: No accuracy error marked by the three raters.

Answer by our human rater: The highlighted source span is indeed translated badly. It contains information that is missing in the translation but can be inferred or is trivial.

Predicted omission error

Source: The automaker is expected to report its quarterly vehicle deliveries in the next **few** days.
Machine translation: Der Autohersteller wird voraussichtlich in den nächsten Tagen seine vierteljährlichen Fahrzeugauslieferungen melden.

Original MQM rating: No related accuracy error marked by the three raters.

Answer by our human rater: The highlighted source span is not translated badly. The words in the span do not need to be translated.

Chinese–English Examples

Predicted addition error

Source: 美方指责伊朗制造了该袭击，并对伊朗实施新制裁。
Machine translation: The US accused Iran of causing the attack and imposed new sanctions **on Iran**.

Original MQM rating (Freitag et al., 2021): No related accuracy error marked by the three raters.

Answer by our human rater: The highlighted target span is not translated badly. No phenomenon that might have caused the prediction was identified.

Predicted omission error

Source: 目前已收到来自俄罗斯农业企业的约50项申请。
Machine translation: About 50 applications have been received from Russian agricultural enterprises.

Original MQM rating: No accuracy error marked by the three raters.

Answer by our human rater: The highlighted source span is indeed translated badly. It contains information that is missing in the translation.

Meaning of highlighted span: 目前 = ‘at present’

Predicted omission error

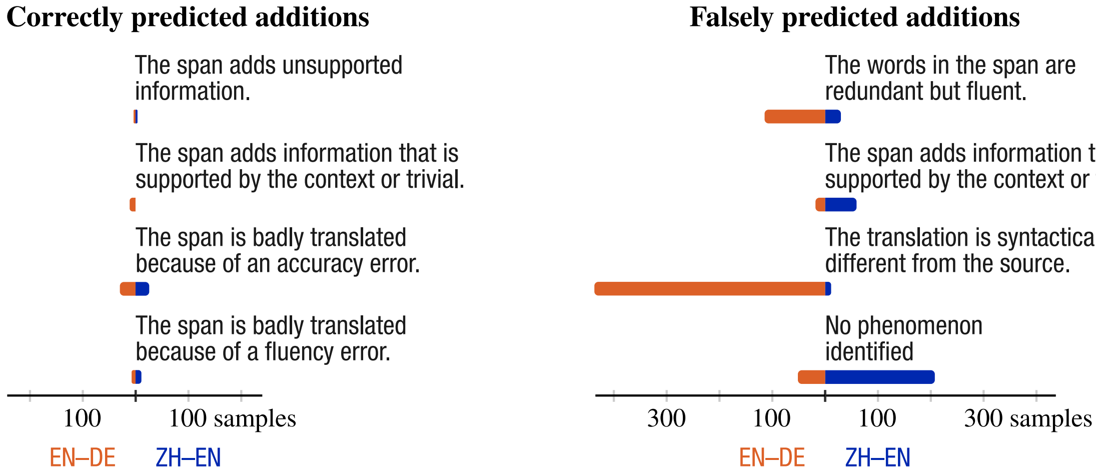
Source: 他说，该系统目前在世界上有很大需求，但俄罗斯军队也需要它，其中包括在北极地区。
Machine translation: He said that the system is currently in great demand in the world, but the Russian army also needs it, including in the Arctic.

Original MQM rating: No accuracy error marked by the three raters.

Answer by our human rater: The highlighted source span is not translated badly. The words in the span do not need to be translated.

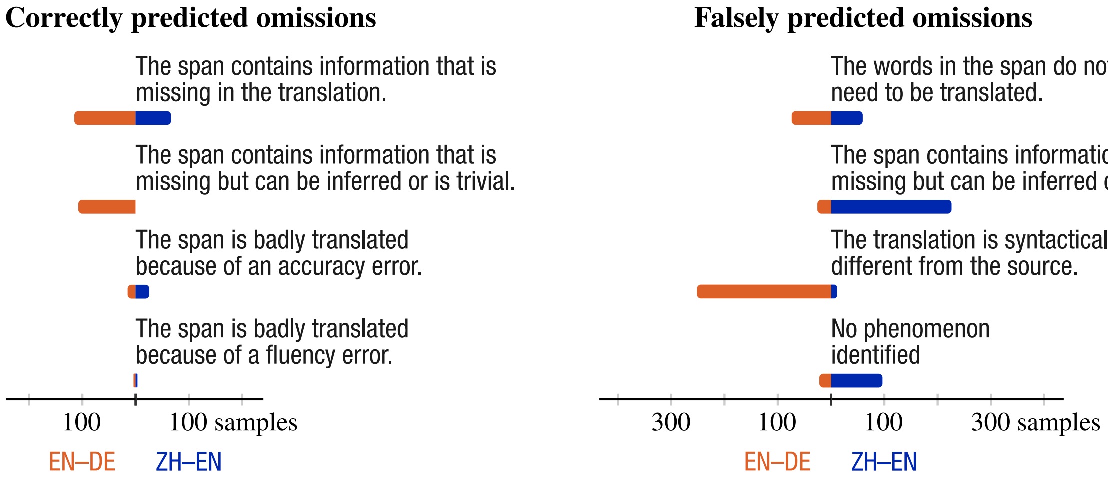
Meaning of highlighted span: 其中 = ‘among’

G Detailed Results of Human Evaluation


Correctly predicted additions

Falsely predicted additions

Figure 3: Results for the human evaluation of predicted addition errors. If human raters answered that the highlighted span in the translation was indeed badly translated, they were offered the four explanation options on the left. Otherwise they chose from the four options on the right.


Correctly predicted omissions

Falsely predicted omissions

Figure 4: Results for the human evaluation of predicted omission errors. If human raters answered that the highlighted span in the source sequence was indeed badly translated, they were offered the four explanation options on the left. Otherwise they chose from the four options on the right.

How Distributed are Distributed Representations? An Observation on the Locality of Syntactic Information in Verb Agreement Tasks

Bingzhi Li and **Guillaume Wisniewski** and **Benoît Crabbé**
Université de Paris, LLF, CNRS
75 013 Paris, France
bingzhi.li@etu.u-paris.fr
{guillaume.wisniewski,benoit.crabbe}@u-paris.fr

Abstract

This work addresses the question of the localization of syntactic information encoded in the transformers representations. We tackle this question from two perspectives, considering the object-past participle agreement in French, by identifying, first, in which part of the sentence and, second, in which part of the representation syntactic information is encoded. The results of our experiments using probing, causal analysis and feature selection method, show that syntactic information is encoded locally in a way consistent with the French grammar.

1 Introduction

Transformers (Vaswani et al., 2017) have become a key component in many NLP models, arguably due to their capacity to uncover distributed representation of tokens (Hinton et al., 1986) that are *contextualized*: thanks to a multi-head self-attention mechanism (Bahdanau et al., 2015), a token representation can, virtually, depend on the representations of all other tokens in the sentence, and transformers are able to learn a weighting to select which tokens are relevant to its interpretation.

Many works (Rogers et al., 2020) strive to analyze the representations uncovered by transformers to find out whether they are consistent with models derived from linguistic theories. One of the main analysis methods is the long-distance agreement task popularized by Linzen et al. (2016), which consists in assessing neural networks ability to predict the correct form of a token (e.g. a verb) in accordance with the agreement rules (e.g. its subject). This method has been generalized to other agreement phenomena (Li et al., 2021) and other languages (Gulordava et al., 2018). The concordant conclusions of all these experiments show that transformers are able to learn a 'substantial amount' of syntactic information (Belinkov and Glass, 2019).

If the method of Linzen et al. (2016) makes it possible to show that syntactic information is encoded in neural representations, it does not give any indication on its localization: it is not clear whether the syntactic information is distributed over the whole sentence (as made possible by self-attention) or only in a way consistent with the syntax of the language, i.e. only in the tokens involved in the agreement rules.

This work addresses the question: *where* the syntactic information is encoded in transformer representations.[1] We approach this question from two perspectives, considering the object-past participle agreement in French (Section 2). First, in Section 3, using probing and counter-factual analysis, we try to identify the tokens in which syntactic information is encoded in order to find its localization within the sentence. Second, in Section 4, using a feature selection method, we study the localization of syntactic information within the contextualized representation of tokens.

2 The Object-Participle Agreement Task

Task We evaluate the capacity of transformers to capture syntactic information, by considering the object-past participle agreement in French object relatives. This task consists in comparing the probabilities a language model assigns to the singular and plural forms of a past participle given the beginning of the sentence. The probability of a past participle form is conditioned on all the words in the *prefix* (the words from the beginning of the sentence up to the antecedent ; see Figure 1 for an example) and the *context* (the words from the antecedent up to and excluding the past participle). Following Linzen et al. (2016) the model is considered to predict the agreement correctly if the form with the correct number has a higher probability

[1] Probing datasets and code available at https://gitlab.huma-num.fr/bli/syntactic-info-distribution

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 501 - 507
May 22-27, 2022 ©2022 Association for Computational Linguistics

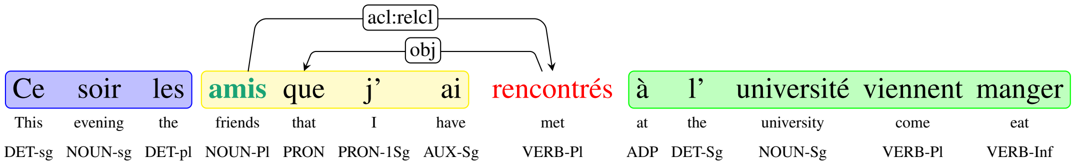

acl:relcl
obj
Ce soir les amis que j' ai rencontrés à l' université viennent manger
This evening the friends that I have met at the university come eat
DET-sg NOUN-sg DET-pl NOUN-Pl PRON PRON-1Sg AUX-Sg VERB-Pl ADP DET-Sg NOUN-Sg VERB-Pl VERB-Inf

Figure 1: Example of object-past participle agreement in French object relatives. Dependencies between the target verb (in red) and the tokens involved in the agreement rules using the Universal Dependencies annotation guidelines are also shown. The *prefix* is represented in blue, the *context* in yellow and the *suffix* in green. To predict the past participle number, a human is expected to extract number information from the object relative pronoun *(que)* that gets it from its antecedent (*amis* in bold green).

than the form with the incorrect number.

Contrary to the classical subject-verb agreement task (Linzen et al., 2016), the French object past participle agreement involves a filler-gap dependency and the target past participle has to agree with a noun that is never adjacent to it. In our case, it features a syntactic structure that allows us to highlight the way the information is distributed in the sentence (§3.1).

Figure 1 gives an example of the sentences considered here. It involves sentences whose verb is in the compound past *(passé composé)*, a tense composed of an auxiliary and the past participle of the verb. What part of the speech (i.e. the subject, the object or no agreement) the compound verbs must agree with depends on the auxiliary verb used. When the past participle is used with the auxiliary *avoir*, it has to agree in number[2] with its direct object when the latter is placed before it in the sentence. This is notably the case for object relatives considered here, in which the direct object is the relative pronoun *que*, whose number information is the same as its antecedent (even if its morphology–*que*, is the same in singular and plural). To correctly agree the past participle in object relatives, it is therefore necessary to identify the object relative pronoun, its antecedent and the auxiliary.

Experimental Setting We reuse the dataset of Li et al. (2021): they have extracted, with simple heuristics a set of 68,497 such sentences after having automatically parsed the Gutenberg corpus with a BERT based dependency parser (Grobol and Crabbé, 2021).

The experiments are carried out with the incremental transformer designed by Li et al. (2021), which was trained on 80 million tokens of French Wikipedia, and has 16 layers and 16 heads. Word embeddings are of size 768. This model is able to predict 93.5% of the past participle agreement, a result that allows these authors to conclude that syntactic information is encoded in the representations.

3 Is Syntactic Information Locally or Globally Distributed in the Sentence?

Results reported in the previous section show that information about the number of the past participle is encoded in the token representations but they do not allow to identify which tokens have been used to predict the correct form of the past participle. In this section, we first identify, using linguistic probes, the tokens in which syntactic information is encoded and then, with a causal analysis, the tokens on which transformers mainly rely to predict the form of the past participle.

3.1 Probing Experiments

In a first set of experiments, we propose to use linguistic probes to better identify **where in the sentence** the information about the number of the past participle is encoded. A probe is a classifier trained to predict linguistic properties from the language representations: achieving high accuracy at this task implies that these properties were encoded in the representation (Hewitt and Manning, 2019).

More precisely, we label each sentence of our dataset with the number of the target verb (i.e. singular or plural) and consider the task of predicting this label from each token representation of the sentence. We trained one logistic regression classifier per category of word[3] considering 80% of the examples as training data and the remaining 20% as test set.

[2] The past participle must agree in number *and* in gender. For clarity, we will only consider agreement in number.

[3] All classifiers are implemented with the Scikit-Learn library (Pedregosa et al., 2011). See detailed description in Section A of the appendix.

	Accuracy		
	correct predictions	wrong predictions	overall
prefix	60.2%$_{\pm 0.3}$	51.6%$_{\pm 0.5}$	59.4%$_{\pm 0.3}$
context	94.6%$_{\pm 0.9}$	83.9%$_{\pm 1.4}$	94.4%$_{\pm 1.1}$
suffix	72.2%$_{\pm 2.1}$	62.1%$_{\pm 2.2}$	71.6%$_{\pm 2.1}$

Table 1: Mean probing accuracies across different sentence parts (see Figure 1) on two subsets, which differ with respect to whether the transformers correctly or incorrectly predicted the number of the past participle.

Table 1 reports the average accuracy achieved by our probes on different parts of the sentence. We observe that the past participle number information is essentially encoded *locally* within the tokens of the *context* and is not represented uniformly across all the subsequent tokens of the sentence as observed by Klafka and Ettinger (2020).

Indeed, as expected,[4] in the *prefix* (before the antecedent) the performance of the probe mainly reflects the difference between the prior probabilities of the two classes.[5] By contrast, the accuracy becomes high when the tokens of the *context* are considered as input features of the probe, showing that the information required to predict the correct past participle form is spread over all tokens between the antecedent (where the number of the past participle is specified) and the past participle (where the information is 'used'). It is quite remarkable that, as soon as the past participle has been observed and the information on the number of the antecedent is no longer useful, the token representations no longer encode it: in the *suffix* the probe accuracy drops sharply even if it remains better than that observed in the *prefix*. This result contradicts also, at least partially, the observation of Wisniewski et al. (2021) which shows that in a neural translation system, gender information is distributed all over the source and target representations. It should however be noted that this experiment deals with a different kind of information and only considers sentences following a very simple pattern.

To get a more accurate picture of how the number information is distributed within the *context*, we focus on a specific sentence template with a fixed six-word *context*: we only consider sentences in which the antecedent is separated from the relative pronoun by a prepositional phrase made of a preposition and a noun as in the following example:

(1) ... magasin d' habits qu' ils ont vu ...
... store of clothes that they have seen...
... ANTEC-SG ADP NOUN-PL QUE PRON-PL AUX-PL PP-SG ...

This pattern (1,940 sentences) represents 3% of the examples of the original dataset. Note that in these sentences the embedded noun between the antecedent of the object pronoun and the target verb can be an *attractor* noun , i.e. a noun with misleading agreement feature. We trained and tested a separate logistic regression classifier for each position as illustrated by the x-axis labels in figure 2.[6]

We plot in figure 2 the average probing accuracy at different positions of this pattern. In the *prefix* (i.e. b-positions) the probe accuracy is low, except for the position just before the antecedent, which often corresponds to determiners or adjectives that have to agree in number with the antecedent. On the contrary, in the *context*, the predictions of the probe are almost perfect, even when we are probing tokens marked with a number information that is not necessarily related to the number of the past participle (e.g. the auxiliary or the attractor). Accuracy in the *suffix* drops quickly as we move away from the past participle, especially in the presence of an attractor. These observations confirm that the number information is not distributed over all tokens in the sentence as made possible by the self-attention mechanism.

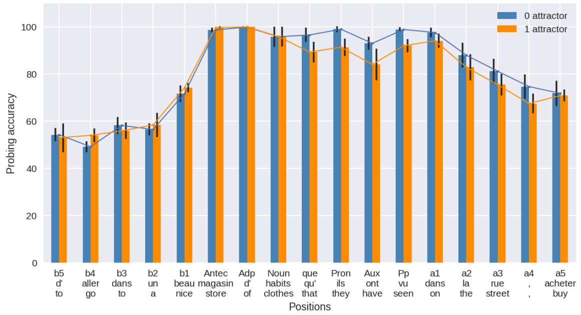


Figure 2: Mean probing accuracy at each position of the six-word *context* pattern. The `bI` (resp. `aI`) position denotes the I-th token before (resp. after) the pattern. An attractor occurs at position *Noun* for 1-attractor subset and the agreeing past participle at position *Pp*.

[4] Recall that we are considering an incremental model in which token representations can only depend on the preceding tokens. The following tokens are masked.

[5] In the dataset, 65% of the past participles are singular.

[6] Note that for purpose of clarity, the plot includes tokens of an example sentence. The results are mean accuracies across all test sentences with three different train/test splits.

Subset	Size (in sentences)	Original	Mask *context* except `Antec que Aux`	Mask `Antec`	Mask `que`	Mask `Antec+que`
Overall	68,200	$93.6\%_{\pm 1.2}$	$85.3\%_{\pm 3.1}$	$84.0\%_{\pm 2.0}$	$79.0\%_{\pm 1.0}$	$76.6\%_{\pm 0.7}$
0 attractor	59,915	$95.4\%_{\pm 0.9}$	$87.3\%_{\pm 3.0}$	$87.5\%_{\pm 1.7}$	$82.9\%_{\pm 0.9}$	$81.3\%_{\pm 0.6}$
1 attractors	7,090	$82.8\%_{\pm 2.5}$	$71.3\%_{\pm 3.9}$	$61.1\%_{\pm 4.2}$	$53.3\%_{\pm 1.7}$	$44.6\%_{\pm 1.4}$
2 attractors	1,195	$71.4\%_{\pm 3.3}$	$68.3\%_{\pm 4.8}$	$47.0\%_{\pm 4.2}$	$36.4\%_{\pm 2.1}$	$27.2\%_{\pm 1.4}$

Table 2: Mean accuracies before and after different masking interventions, based on prediction difficulty measured by the number of attractors

3.2 Causal intervention on attention

As it stands, we observe that number information is encoded essentially in the *context* part of sentences. Now we test **which** tokens are responsible for providing the number information used to choose the past participle form. To do so, we design a causal experiment in which we mask some tokens of the *context* to better figure out their role in models decision.

Masking Tokens in Self-Attention Computation Self-attention is a core component of transformers. In our causal analysis we mask some token representations in the *context* to the self-attention layer. By design, incremental transformers are already masking the end of the sentence with a boolean mask to prevent a token representation to attend to the future tokens. We extend this mechanism to mask, when computing the past participle representation, additional tokens from the sentence prefix such as the antecedent and the relative pronoun.

This intervention allows us to suppress direct access to some tokens such as the antecedent (and thus its number) when building the past participle representation, even if the latter can still access them indirectly: it indeed relies on all other tokens in the sentence for which the mask is kept unchanged. It is then possible, as featured in ablation experiments, to compare performances on the agreement task with and without intervention to evaluate whether the representation of a given token has a direct impact on the prediction of the past participle form.

Results Table 2 reports the accuracy on the object-past participle agreement task when some of the tokens in the context are masked. Accuracies are broken down by the number of attractors found in the *context*, a proxy to the difficulty of the prediction (Gulordava et al., 2018). Results show that masking either of the tokens involved in the agreement rule (i.e. the relative pronoun *que* or the antecedent) strongly degrades prediction performance. On the contrary, masking all tokens in *context* except these two and the token before the target verb (generally the auxiliary) has a limited impact on models performance, especially for the most difficult case. This suggests that transformers learn representations that are consistent with the French grammar: the model relies mainly on the same tokens as humans to choose the correct form of the past participle.

4 Probing Representations Components

Experiments reported in the previous section show that syntactic information is locally encoded in the *context*. In this section, we address the question of finding **where** this information is encoded **within the transformers representation**. To that end, we repeat the probing experiment on *context* token representations of §3.1 with an ℓ_1 regularized logistic regression (Tibshirani, 1996). The resulting probe is thus constrained to minimize the number of features used to perform accurate predictions. Given the probe objective function $\sum_{i=1}^{n} - \log P(y_i|\mathbf{x}_i; \mathbf{w}) + \frac{1}{C}||\mathbf{w}||_1$ to minimize, we first determined the lowest bound for C such that the feature coefficients are guaranteed not to be all zeros, from which, we increase C evenly on a log space (i.e. decrease the regularization strength).

Results Figure 3 reports the regularization path of the probing classifier. It shows that number information can be extracted with high accuracy (90.1%) solely from a very small number of dimensions, namely 90. Increasing the number of dimensions (by decreasing the regularization strength) only results in a small improvement of model quality: the probe achieves an accuracy of 94.8% when all features are considered. Interestingly, when removing the 90 features selected by the ℓ_1 regularization from the representation, a probe trained on the remaining features still achieve a very good accuracy of 93.8%, suggesting that the number information is encoded in a redundant way in the contextualised representations.

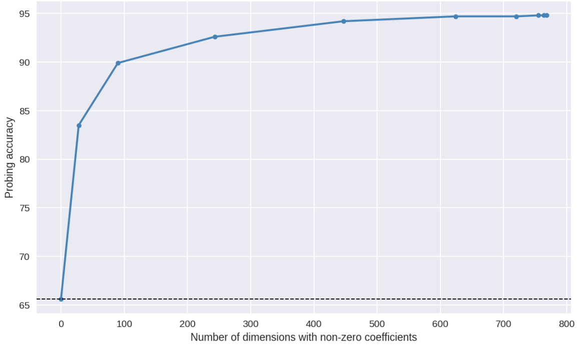


Figure 3: Feature selection by ℓ_1-logistic regression: probing accuracy of all *context* token representations

5 Discussion and conclusion

To understand how syntactic information is encoded and used in transformers-based LM, we carried out three sets of experiments considering the French object-past participle agreement task. First, our probing experiments uncovered clear evidence of a local distribution of number information within the *context* tokens, even though the self-attention mechanism allows this information to be spread all over the sentence. Second, our masking intervention on attention shows a causal link between linguistically motivated tokens and the model's decision, suggesting that transformers process French object-past participle agreement in a linguistically-motivated manner. Finally, we used a ℓ_1 feature selection method to study the localization of number information within contextualized representations and found that while this information is encoded in a small amount of highly correlated dimensions, it is also fuzzily encoded in a redundant way in the remaining dimensions.

Our work is a first step towards a better understanding of the inner representations of LM. Designing new probes, supported by causal analysis and involving a wider range of languages, could improve our understanding of such models. In particular, our observation about the linguistically motivated distribution of syntactic information in transformers representations could be extended to other linguistic phenomenon and languages.

Acknowledgments

We sincerely thank the reviewers and Program Chairs for their careful reviews and insightful comments, which are of great help in improving the manuscript. This work was granted access to the HPC resources of French Institute for Development and Resources in Intensive Scientific Computing (IDRIS) under the allocation 2020-AD011012282 and 2021-AD011012408 made by GENCI. We would also like to gratefully acknowledge support from the Labex EFL, "Empirical Foundations in Linguistics" (ANR-10-LABX-0083)

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Loïc Grobol and Benoit Crabbé. 2021. Analyse en dépendances du français avec des plongements contextualisés (French dependency parsing with contextualized embeddings). In *Actes de la 28e Conférence sur le Traitement Automatique des Langues Naturelles. Volume 1 : conférence principale*, pages 106–114, Lille, France. ATALA.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Geoffrey E. Hinton, James L. McClelland, and David E. Rumelhart. 1986. Distributed representations. In *Parallel distributed processing: Explorations in the microstructure of cognition. Volume 1: Foundations*.

Josef Klafka and Allyson Ettinger. 2020. Spying on your neighbors: Fine-grained probing of contextual embeddings for information about surrounding words. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4801–4811, Online. Association for Computational Linguistics.

Bingzhi Li, Guillaume Wisniewski, and Benoit Crabbé. 2021. Are Transformers a modern version of ELIZA? Observations on French object verb agreement. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4599–4610, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

R. Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288.

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

Guillaume Wisniewski, Lichao Zhu, Nicolas Bailler, and François Yvon. 2021. Screening gender transfer in neural machine translation. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 311–321, Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Probing classifiers

We used a set of logistic regression classifiers to investigate the way the syntactic information is distributed inside the sentences. Each sentence are divided into three parts: *prefix*, *context* and *suffix*, as described in Figure 1. The input for all classifiers are the contextualized token representations built by our pre-trained transformers. We trained one classifier per category of word and per part of the sentences to predict whether the token representation is singular or plural, forcing each probing classfier to specialise on PoS-specific representations of long-distance agreement information. To ensure a fair comparison across parts of sentences, we eliminated the following tokens of PoS tags with less than 100 occurrences: SYM, SCONJ, INTJ, PART, PART and X. Therefore, we have in total 11 categories of tokens in each part of the sentences, resulting in 11*3 probing classifiers, and each classifier is trained with three train/test splits(i.e. random_state $= 0, 20$ and 42). The averaged results is reported in table 1 of the paper. The detailed results per category of word is in figure 4 below.

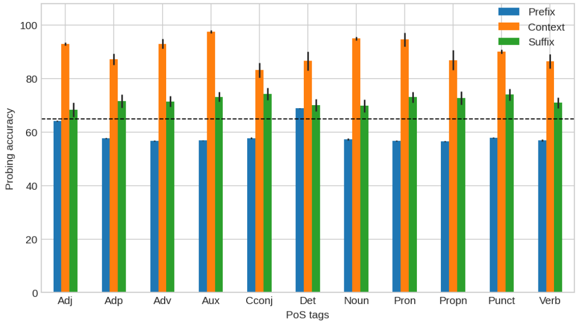


Figure 4: Probing accuracy based on tokens PoS tags and their positions in the sentences, from left to right: *prefix*, *context*, *suffix*

Machine Translation for Livonian: Catering to 20 Speakers

Matīss Rikters$^{\tau}$, Marili Tomingas$^{\tau}$, Tuuli Tuisk$^{\tau\lambda\kappa}$, Valts Ernštreits$^{\lambda}$, Mark Fishel$^{\tau}$

$^{\tau}$ University of Tartu
{matiss.rikters, tuuli.tuisk, marili.tomingas, fishel}@ut.ee
$^{\lambda}$ University of Latvia
valts.ernstreits@lu.lv
$^{\kappa}$ University of Copenhagen

Abstract

Livonian is one of the most endangered languages in Europe with just a tiny handful of speakers and virtually no publicly available corpora. In this paper we tackle the task of developing neural machine translation (NMT) between Livonian and English, with a two-fold aim: on one hand, preserving the language and on the other – enabling access to Livonian folklore, lifestories and other textual intangible heritage as well as making it easier to create further parallel corpora. We rely on Livonian's linguistic similarity to Estonian and Latvian and collect parallel and monolingual data for the four languages for translation experiments. We combine different low-resource NMT techniques like zero-shot translation, cross-lingual transfer and synthetic data creation to reach the highest possible translation quality as well as to find which base languages are empirically more helpful for transfer to Livonian. The resulting NMT systems and the collected monolingual and parallel data, including a manually translated and verified translation benchmark, are publicly released via the OPUS corpora collection and Huggingface model repository.

1 Introduction

Many state-of-the-art natural language processing tasks have reached admirable quality on languages with abundant linguistic resources (Vaswani et al., 2017; Conneau et al., 2018; Devlin et al., 2019). Furthermore, some neural language models and translation systems have been created for 100 and more languages (e.g. Conneau et al., 2020; Fan et al., 2021). However smaller, less or not at all spoken languages continue to struggle not only in terms of applicable computational approaches, but more critically - in terms of usable resources for training natural language processing (NLP) models or even just linguistic exploration.

In this paper we set the goal of developing machine translation between English and Livonian. Currently there are just over 20 fluent speakers of the language (Ernštreits, 2016). Although some digital linguistic resources exist for Livonian (including a dictionary with example sentences and a written monolingual corpus, Ernštreits, 2016), there is virtually no open parallel corpora between English and Livonian, with the single exception of 35 parallel sentences in the OPUS Tatoeba corpus (Tiedemann, 2020).

At the same time, cross-lingual transfer learning has recently helped improve the performance of several low-resource NLP tasks with the support of related languages (e.g. Conneau et al., 2018; Hu et al., 2020). This also includes zero-shot translation (Johnson et al., 2017), the ability of multilingual NMT systems to translate between seen languages that were not represented in the parallel training data as a pair. The case of Livonian is especially interesting in this regard, as there are two different sources of such support: on one hand, it is a Uralic language, closely related to Estonian and Finnish. On the other hand, Livonian has taken part in forming Latvian language and Livonian speakers have historically co-existed side-by-side with Latvian speakers. As a result of mutual influence these two languages also share a number of grammatical, lexical and orthographic similarities.

Our main contributions are two-fold. First, we collected the majority of digitally available translation examples including Livonian into a small parallel corpus (just over 10000 sentence pairs) of mostly Livonian-Latvian and Livonian-Estonian sentence translations with very few (1000) Livonian-English examples. In order to create a clean benchmark for evaluating translation quality we selected a portion (about 10%) of this corpus and had it manually translated into Latvian/Estonian/English so that each sentence would

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 508 - 514
May 22-27, 2022 ©2022 Association for Computational Linguistics

Source	LIV-ENG	LIV-EST	LIV-LAT
Dictionary examples	–	10 690 / 44 854 / 44 499	10 690 / 44 854 / 44 975
Latvian constitution	686 / 11 198 / 15 499	719 / 11 454 / 10 314	719 / 11 454 / 11 002
JEFUL abstracts	–	187 / 2 878 / 2 846	176 / 2 723 / 3 434
Facebook posts	231 / 2 759 / 3 656	8 / 124 / 122	232 / 2 744 / 2 738
livones.net texts	169 / 2 741 / 3 660	92 / 1 969 / 1 867	333 / 4 449 / 4 433
Stalte ABC book	–	1 340 / 9 382 / 9 195	1 340 / 9 382 / 9 398
Trilium, poetry book	–	222 / 3 543 / 3 321	223 / 3 512 / 3 539
Eduard Vääri book	–	877 / 10 337 / 9 763	–
Total	**1 086 / 16 698 / 22 815**	**14 135 / 84 541 / 81 927**	**13 713 / 79 118 / 79 519**

Table 1: Total data size for the collected parallel LIV<->ENG/EST/LAT data. Each cell includes the sentence count, and word count for Livonian and the other language.

have all four manually verified translations.[1]

The second half of our work focuses on neural machine translation (NMT, Vaswani et al., 2017), mainly targeting Livonian↔English. We explore several options of coping with the extremely low-resource settings and use Estonian and Latvian for cross-lingual transfer. Our experiments answer the following research questions:

1. Can we achieve machine translation for Livonian↔English at a usable level?

2. Which base language suits better for serving as base for cross-lingual transfer to Livonian, Estonian or Latvian?

3. Does zero-shot multilingual translation deliver better translation quality than pivot-translation through Estonian or Latvian?

Next we briefly describe the Livonian Language in Section 2, then introduce the collected parallel and monolingual data in Section 3. Section 4 provides the details of our NMT experiments and Section 5 concludes the paper.

2 The Livonian Language

Livonian (ISO 639-3: `liv`) is a Finnic language indigenous to Latvia and belonging to the Uralic language family. During the 12th century Livonian was spoken across great territories in Latvia around the Gulf of Riga. Over time, Livonian areas gradually became Latvian-speaking. In the 19th century, Livonian still had approximately 2500 speakers, by the mid-20th century around 1500 speakers. Nowadays Livonian is listed in UNESCO's Atlas of the World's Languages in Danger as a critically endangered language (Moseley, 2014). According to the 2011 census, there are 250 Livonians in Latvia. Although there are just over 20 people who can speak the language, the Livonian community is active in preserving and developing the Livonian heritage (Ernštreits, 2016) and language plays a key role in this process (Ernštreits and Kļava, 2020).

The Livonian language developed in the contact area of Baltic and Finnic languages. Livonian and Latvian share a similar geographical location over a prolonged period of time, as a result of which they both contain traces of contact. Next to other loanwords, the Livonian loanword strata consists of words borrowed from Latvian (Suhonen, 1973; Winkler, 2014) and vice versa. The most obvious Latvian influence on Livonian grammar is found in the Livonian case system (Ernštreits and Kļava, 2014). Livonian has the prosodic characteristics typical of a Finnic language such as word-initial stress and the phonological opposition of short and long phoneme duration. It is the only Finnic language that differentiates lexical tones – the plain tone and the broken tone or *stød* – and therefore shares similar characteristics with Latvian as well as Danish (Tuisk, 2016).

3 Collected Data

The first step in developing (supervised) machine translation is collecting parallel data. While there was no pre-existing open parallel corpus with Livonian, we used all the possible sources of translations. This was limited to already digital resources, future work might include texts extracted by scanning older books and other materials.

[1] Translation from Livonian was a too rare and expensive service, thus we resorted to translating from one of the other three languages and instead had Livonian speakers check the results for meaning correspondence afterwords.

	LV→EN	**ET→EN**	**ETLV→EN**	**EN-ET-LV**	**Google**	**Neurotolge**
ET		30.91	28.42	24.17	34.38	29.91
LV	25.18		25.26	20.77	31.54	25.92
LIV	2.20	3.22	2.66	13.29	-	-
Tuned						
LIV→EN	3.19	5.59	5.39	14.69	-	-
EN→LIV	-	-	-	8.59	-	-

Table 2: Results from machine translation experiments for translating into English. The source languages are listed in the first column and different models for translation are in each further column. We also compared ET/LV→EN translations of our evaluation set using Google Translate[7] and Neurotõlge[8] online translation services.

The main sources of data included Livonian-Latvian as well as Livonian-Estonian translations. Thus we use these two languages as base for cross-lingual transfer and e.g. leave Finnish out, as there was no data for it.

The sources of data included:

- the Constitution of the Republic of Latvia, translated into 9 languages, including Livonian, Estonian and English,
- a database of dictionary entries, phrases and example sentences from the University of Latvia Livonian Institute's website[2], with example sentences in Livonian, Estonian and Latvian
- the Livonian Institute's Facebook page posts, partially parallel between our 4 languages
- books (Stalte, 2011; Kurs and et al., 2016; Ernštreit et al., 2020) with prefaces and content in Livonian-Estonian or Livonian-Latvian
- and abstracts from the Journal of Estonian and Finno-Ugric Linguistics' (JEFUL) Special Issues on Livonian Studies (2014, 2016, 2018) in Livonian, Estonian and English.

Concerning sentence alignment, the dictionary examples consisted of already aligned Livonian sentences. We aligned the rest of the data manually with the help of language experts – first on paragraph level, then on sentence level. The resulting amount of sentences in the resulting dataset is shown in Table 1.

We separated balanced portions of development (503 sentences) and evaluation (749 sentences) splits from the full dataset. The splits are balanced in terms of the original source of the texts to resemble proportions from the remaining training data. We hired professional translators to create translations for any missing parts so that these splits would be parallel between all four languages. We further turned to experts of the Livonian language to make sure that the newly created translations truly convey the meaning of the original text as a quality control measure. The resulting benchmark and the whole corpus is published in the OPUS collection.[3] We also share the final translation model[4] after four iterations of backtranslation.

4 Machine Translation Experiments

Having just over $10,000$ parallel examples constitutes extremely low-resource settings for neural machine translation. Added to this, the number of monolingual Livonian sentences (about $40,000$) is also too small for approaches like unsupervised machine translation (Artetxe et al., 2018; Lample et al., 2018).

We implement the support of neighboring and related languages (Estonian and Latvian) via multilingual machine translation (Johnson et al., 2017). As a first step the model is pre-trained with the larger languages (Estonian, Latvian, English) and then used as base for following experiments.

We also perform iterative back-translation (Pinnis et al., 2018) to make use of the large amounts of monolingual news data in EN/ET/LV, and our limited amount of monolingual data in LIV. We translate the 40k LIV sentences and different batches of 200k sentences from the other languages into all directions, filter the translations using simple heuristic filters (Rikters, 2018), and use a mix of all back-translated data with an equal amount of random clean parallel data (including all data involving Livonian) to fine-tune the base model.

[2] www.livones.net/

[3] https://opus.nlpl.eu/liv4ever.php

[4] https://huggingface.co/tartuNLP/liv4ever-mt

	Base	**Tuned**	**BT1**	**BT2**	**BT3**	**BT4**
ET-EN	24.17	23.68	23.97	24.80	25.05	**26.17**
LV-EN	20.77	18.90	19.29	20.95	20.52	**21.53**
LIV-EN	13.29	14.69	16.19	17.41	18.15	**19.01**
EN-ET	17.00	16.87	18.58	19.37	18.95	**19.48**
LV-ET	18.38	19.55	19.72	19.93	20.68	**22.38**
LIV-ET	15.08	17.76	20.05	21.61	21.78	**23.05**
EN-LV	16.57	17.94	17.17	19.58	19.49	**20.85**
ET-LV	18.51	21.16	20.92	21.01	21.96	**23.44**
LIV-LV	15.05	17.55	21.25	22.99	23.68	**25.24**
EN-LIV	4.19	8.59	9.96	10.49	10.88	**11.03**
ET-LIV	4.01	13.00	14.43	15.24	16.09	**16.49**
LV-LIV	4.84	13.67	15.18	16.25	16.77	**17.65**

Table 3: Results in BLEU scores from the model at each training iteration translating in all translation directions.

4.1 Technical Setup

We used FairSeq (Ott et al., 2019) to train transformer architecture models with 6 encoder and decoder layers, 8 transformer attention heads per layer, word embeddings and hidden layers of size 512, dropout of 0.3, maximum sentence length of 128 symbols, and a batch size of 1024 words. All models were trained until they reached convergence (no improvement for 10 checkpoints) on development data. We used Sentencepiece (Kudo and Richardson, 2018) to create shared vocabularies of size 25,000, and SacreBLEU[5] (Post, 2018) to generate BLEU scores (Papineni et al., 2002) for translations.

Base models were trained on LV→EN, ET→EN, ET+LV→EN data, and a multilingual model using the tagged approach (Johnson et al., 2017) for translating in all directions between EN/ET/LV languages. The base models were then used as initialization for tuning on Livonian-English parallel data.

For training the base models we used all available parallel data from Opus (Tiedemann and Nygaard, 2004). To facilitate further use of the base models for tuning on Livonian data, all Livonian sentences were used in addition to other data when creating the shared vocabularies. Finally, we used the highest-scoring tuned model to perform performed backtranslation on the monolingual LIV data to generate additional training data for training the final models.

4.2 Results

Table 2 shows the results of MT experiments. All BLEU scores are calculated for translations of our evaluation set. We compare the base single direction MT models to our multidirection model, as well as online translations from Google Translate[6] and Neurotolge[7] to evaluate performance from ET and LV into EN. While the multilingual model was noticeably weaker, the others hold comparable results to the online systems. However, when attempting to perform zero-shot translation from LIV into EN, ET→EN outperforms LV→EN (3.22 vs. 2.20), and the multilingual model achieved a very respectable BLEU score 13.29.

We then turned to tuning each of these models with LIV-EN data mixed 1:1 with a random equal amount of the original training data for each of the models. In the case of the multilingual model, we also added LV/ET-LIV data to the mix. This improved all scores by 1-3 BLEU points, but the multilingual model remained on top with 14.69 for LIV→EN. In order to perform backtranslation models for both directions are required, so we scored the tuned multilingual model on the EN→LIV data as well, reaching 8.59 BLEU.

For comparison we also used the same tuned multilingual model to perform pivotal translation by first translating into ET or LV and then into the desired target language. In all four cases the pivot translation quality dropped when compared to direct translation by the same model, so we did not further pursue this line of experiments. An

[5] Case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.5.1

[6] https://translate.google.com - accessed in Nov. 2021

[7] https://neurotolge.ee - accessed in Nov. 2021

	LIV-EN	EN-LIV
Facebook	19.28	13.55
Livones.net	19.67	15.91
Dictionary	7.73	10.60
Trilium	19.88	14.50
Stalte	13.88	9.47
JEFUL	8.02	5.10
Satversme	24.49	7.69

Table 4: Detailed experiment results in BLEU scores, split by the source of data from the last run of back-translation (BT4).

interesting observation, was that pivoting through ET achieved a higher BLEU score than LV when translating into EN (13.66 vs. 11.24), but slightly lower when translating into LIV (7.99 vs. 8.56).

Results for four rounds of BT iterations are compiled in Table 3. The model clearly improves not only in the main language pair of EN↔LIV, but in all other translation directions as well.

To answer the research questions, posed in the introduction, it seems that the resulting translation quality is still far from being usable. Comparisons between the base languages have shown slight preference towards Estonian over Latvian. Pivot-translation trough Estonian or Latvian underperforms direct Livonian↔English translation trained in a zero-shot / few-shot manner.

4.3 Detailed Analysis

Table 4 shows BLEU scores of the separate parts of the evaluation corpus. Since most of the training data for EN-LIV comes from Satversme (Latvian Constitution), it is very clear why that part scores higher than others. The dictionary entries are overall far shorter in length than the other parts and often consist of few-word phrases, making them unfavorable to BLEU by definition.

The posts from Facebook and Livones.net are more general in their language and therefore more similar to data from the training set. However, the Trilium and Stalte books are written in a more literary language, making them slightly more challenging to translate. Finally, the very domain-specific part from JEFUL abstracts seems to be the most difficult to translate into English.

5 Conclusion

In this paper we presented a novel dataset for the highly endangered Livonian language, which can be useful for machine translation, language modelling and many other natural language processing and computational linguistic research tasks.

In our experiments we show how far one can get in training modern machine translation models with very scarce data, and which languages are more suitable for transfer learning when working with Livonian data. While perhaps not being usable as-is in any kind of production scale, the achieved final BLEU scores of 19.01 for Livonian→English and 11.03 for English→Livonian show that some transfer of meaning can still be achieved with the currently available resources.

In the future we are planning to experiment with cross-lingual transfer from other languages, like the resource-rich Finnish as well as resource-poor Finno-Ugric languages like Võru and Sami (Tars et al., 2021). Given the limited amount of existing monolingual Livonian data, generating synthetic Livonian data with other means besides back-translation might be helpful: for example, forward-translation or using GPT-like language models.

Finally, work on the already collected Livonian monolingual and parallel data is ongoing at the Institute of the Livonian Language. Adding English translations to the lexical items and example sentences is an ongoing effort and will evaluate in practice, if the MT systems created as part of the current work can facilitate that. One of the key focuses is also manually verifying the data and making sure the existing corpus contains correct Livonian texts and their translations

Acknowledgements

This work has received funding from the "European Social Fund via IT Academy programme," Estonian Ministry of Education and Research (grant SHVEE21397), the Estonian Research Council (grant GHVEE22112J), and the State Research Programme "Latvian Studies for the Development of a Latvian and European Society" project "Multifunctional dictionary of Livonian" (No. VPP-LETONIKA-2021/2-0002).

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. Unsupervised statistical machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3632–3642, Brussels, Belgium. Association for Computational Linguistics.

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

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

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

Valts Ernštreits. 2016. Livonian in recent years. *Eesti ja soome-ugri keeleteaduse ajakiri. Journal of Estonian and Finno-Ugric Linguistics*, 7(1):257–274.

Valts Ernštreits and Gunta Kļava. 2014. Grammatical changes caused by contact between livonian and latvian. *Eesti ja soome-ugri keeleteaduse ajakiri. Journal of Estonian and Finno-Ugric Linguistics*, 5(1):77–90.

Valts Ernštreits and Gunta Kļava. 2020. Taking the livonians into the digital space. In *Proceedings of the 5th Conference Digital Humanities in the Nordic Countries*, pages 26–37, Riga, Latvia.

Valt Ernštreit, Baiba Damberg, and Karl Pajusalu. 2020. *Trilium 2.0. Līvõ lūolkub. Liivi luulekogu. Lībiešu dzejas izlase.* The International Society of Livonian Friends.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multitask benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Ott Kurs and et al., editors. 2016. *Language and mind of the Livonian people. Eduard Vääri's publications on Livonians and the Livonian language*, volume 74. Eesti Teaduste Akadeemia Emakeele Seltsi toimetised.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, Brussels, Belgium. Association for Computational Linguistics.

Christopher Moseley. 2014. Livonian–the most endangered language in europe? *Eesti ja soome-ugri keeleteaduse ajakiri. Journal of Estonian and Finno-Ugric Linguistics*, 5(1):61–75.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Mārcis Pinnis, Matīss Rikters, and Rihards Krišlauks. 2018. Tilde's machine translation systems for WMT 2018. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 473–481, Belgium, Brussels. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Matīss Rikters. 2018. Impact of Corpora Quality on Neural Machine Translation. In *In Proceedings of the 8th Conference Human Language Technologies - The Baltic Perspective (Baltic HLT 2018)*, Tartu, Estonia.

Kōrli Stalte. 2011. *Jelzi Sõnā: ābēd ja īrgandõks lugdõbrõntõz*. Jemākīel seļtš.

Seppo Suhonen. 1973. Die jungen lettischen lehnwörter im livischen. *Suomalais-ugrilaisen seuran toimituksia*, 154.

Maali Tars, Andre Tättar, and Mark Fišel. 2021. Extremely low-resource machine translation for closely related languages. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 41–52, Reykjavik, Iceland (Online). Linköping University Electronic Press, Sweden.

Jörg Tiedemann. 2020. The tatoeba translation challenge – realistic data sets for low resource and multilingual MT. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1174–1182, Online. Association for Computational Linguistics.

Jörg Tiedemann and Lars Nygaard. 2004. The OPUS corpus - parallel and free: `http://logos.uio.no/opus`. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Tuuli Tuisk. 2016. Main features of the livonian sound system and pronunciation. *Eesti ja soome-ugri keeleteaduse ajakiri. Journal of Estonian and Finno-Ugric Linguistics*, 7(1):121–143.

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

Eberhard Winkler. 2014. Loanword strata in livonian. *Eesti ja soome-ugri keeleteaduse ajakiri. Journal of Estonian and Finno-Ugric Linguistics*, 5(1):215–227.

Fire Burns, Sword Cuts: Commonsense Inductive Bias for Exploration in Text-based Games

Dongwon Kelvin Ryu♠ **Ehsan Shareghi**♠ ♣ **Meng Fang**♡
Yunqiu Xu♢ **Shirui Pan**♠ **Gholamreza Haffari**♠
♠ Department of Data Science & AI, Monash University
♡ Eindhoven University of Technology ♢ University of Technology Sydney
♣ Language Technology Lab, University of Cambridge
firstname.lastname@monash.edu m.fang@tue.nl
yunqiu.xu@student.uts.edu.au

Abstract

Text-based games (TGs) are exciting testbeds for developing deep reinforcement learning techniques due to their partially observed environments and large action spaces. In these games, the agent learns to explore the environment via natural language interactions with the game simulator. A fundamental challenge in TGs is the efficient exploration of the large action space when the agent has not yet acquired enough knowledge about the environment. We propose COMMEXPL, an exploration technique that injects external commonsense knowledge, via a pretrained language model (LM), into the agent during training when the agent is the most uncertain about its next action. Our method exhibits improvement on the collected game scores during the training in four out of nine games from Jericho. Additionally, the produced trajectory of actions exhibit lower perplexity, when tested with a pretrained LM, indicating better closeness to human language. [1]

1 Introduction

Text-based games (TGs) are environments where agents learn to comprehend situations in language and produce decisions in language (Hausknecht et al., 2020; Côté et al., 2018; Narasimhan et al., 2015). Deep Reinforcement Learning lends itself as a natural paradigm to solve TGs due to its ability to learn from unsupervised game playing experience. However, existing RL agents are far away from solving TGs due to their combinatorially large action spaces that hinders efficient exploration (Yao et al., 2020; Ammanabrolu and Hausknecht, 2020).

Ammanabrolu and Riedl (2019); Ammanabrolu and Hausknecht (2020) proposed incorporating a belief knowledge graph (BKG) built from the textual observations to help the agent reason more effectively about observed objects during the gameplay. Most of the recent works neglected linguistic aspects of TGs and focused on the construction and utilisation of BKG (Adhikari et al., 2020; Dambekodi et al., 2020; Xu et al., 2020; Ammanabrolu et al., 2020; Xu et al., 2021). Some exceptions involve developing pre-trained language models (LMs) to propose action candidates for a given observation (Yao et al., 2020), and investigating the relationship between semantic coherence and state representations (Yao et al., 2021).

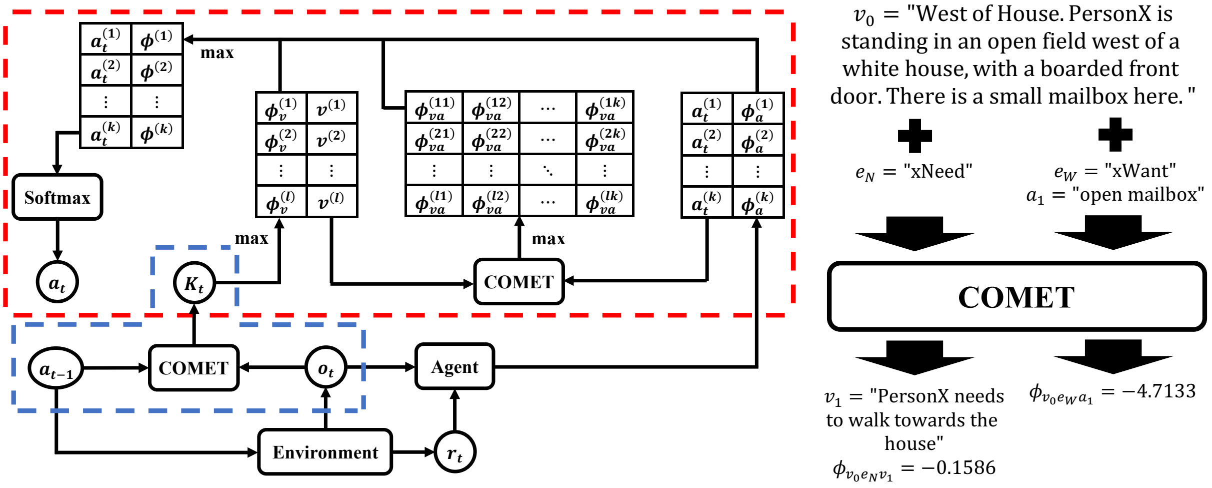
In parallel, it has been argued that recent pretrained LMs capture commonsense factual knowledge about the world (Petroni et al., 2019; Kassner et al., 2021; Meng et al., 2021). More direct attempt in this direction was the commonsense transformer (COMET) which is a LM fine-tuned explicitly on commonsense knowledge graph (CSKG), to explicitly generate commonsense inferences (Bosselut et al., 2019; Hwang et al., 2021). Prior works with commonsense focused on completing BKG using pre-defined CSKG (Murugesan et al., 2020) or dynamic COMET-generated commonsense inferences (Dambekodi et al., 2020). Nonetheless, there is no work on explicitly using commonsense as an inductive bias in the context of exploration for TGs.

To bridge the gap, we propose *commonsense exploration* (COMMEXPL) which constructs a CSKG dynamically, using COMET, based on the state of textual observation per step. Then, the natural language actions are scored with COMET and agent, to re-rank the policy distributions. We refer to this as applying *commonsense conditioning*. However, doing this throughout the whole training is expensive and may not be beneficial as gameplay is not led by commonsense. To rectify this, we propose an *entropy scheduler*, driven by the entropy of the policy distribution, to regulate applying commonsense conditioning.

We demonstrate that our method encourages

[1] Code is available at https://github.com/ktr0921/comm-expl-kg-a2c

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 515 - 522
May 22-27, 2022 ©2022 Association for Computational Linguistics



$a_t^{(1)}$ $\phi^{(1)}$
$a_t^{(2)}$ $\phi^{(2)}$
$a_t^{(k)}$ $\phi^{(k)}$
max
Softmax
a_t
$\phi_v^{(1)}$ $v^{(1)}$
$\phi_v^{(2)}$ $v^{(2)}$
$\phi_v^{(l)}$ $v^{(l)}$
max
K_t
$\phi_{va}^{(11)}$ $\phi_{va}^{(12)}$ $\phi_{va}^{(1k)}$
$\phi_{va}^{(21)}$ $\phi_{va}^{(22)}$ $\phi_{va}^{(2k)}$
$\phi_{va}^{(l1)}$ $\phi_{va}^{(l2)}$ $\phi_{va}^{(lk)}$
max
COMET
$a_t^{(1)}$ $\phi_a^{(1)}$
$a_t^{(2)}$ $\phi_a^{(2)}$
$a_t^{(k)}$ $\phi_a^{(k)}$
a_{t-1}
COMET
o_t
Agent
Environment
r_t
v_0 = "West of House. PersonX is standing in an open field west of a white house, with a boarded front door. There is a small mailbox here."
e_N = "xNeed"
e_W = "xWant"
a_1 = "open mailbox"
COMET
v_1 = "PersonX needs to walk towards the house"
$\phi_{v_0 e_N v_1} = -0.1586$
$\phi_{v_0 e_W a_1} = -4.7133$

Figure 1: (Left) The overall architecture of COMMEXPL. The blue region is the *CSKG Construction* and the red region is *commonsense conditioning*. During *CSKG Construction*, COMET generates CKSG $\mathcal{K}$ given an action-observation pair while it produces node-to-action score given a node-action pair in *commonsense conditioning*. (Right) Example of how COMET works in COMMEXPL: Given a head node and edge, a tail node and its corresponding node-to-node score is generated while for node-to-action score, an action is passed as a desired tail node in COMET. Notations are defined in §2.

the agent to achieve higher game score during the training in four out of nine games in Jericho (Hausknecht et al., 2020). Furthermore, we show our method leads to producing more human-like natural language action. This is measured using the perplexity of the generated actions according to GPT-2 (Radford et al., 2019). We believe that natural language coherency/fluency is a crucial aspect of interactive intelligent agents (e.g. robots and dialogue systems) and hope our promising findings facilitate further developments of methods in this direction.

2 Approach

Notations. Text-based games are modelled as a partially observable Markov decision processes (POMDPs) of a tuple of $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{O}, \Omega, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$, $\mathcal{A}$, Ω denote sets of states, actions, and observations, respectively. Also, $\mathcal{R}$ and γ denote the reward function and the discount factor, while $\mathcal{P}$ and $\mathcal{O}$ denote the transition probabilities and set of conditional observations probabilities, respectively.

The agent requires to map an observation to a state ($\Omega \rightarrow \mathcal{S}$) and produce a policy π. By selecting an action a_t from the policy π, the agent changes current state s_t, receives a reward signal r, receives an observation through transition $\mathcal{P}(s_{t+1}|s_t, a_t)$, and also receives a conditional observation $\mathcal{O}(\Omega_t|s_t)$. The agent learns the policy $\pi_\theta(\boldsymbol{a}|\boldsymbol{o})$ that maximizes the expectation of the cumulative reward function $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$.

2.1 CSKG Construction

Let a CSKG be a graph $\mathcal{K} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of nodes or vertices and $\mathcal{E}$ is a set of edges. The root node of CSKG requires to carry adequate information about the gameplay, so we amend the input to be the same format as how COMET is trained on, $\boldsymbol{v}_0 =$ "I "$+\boldsymbol{a}_{t-1}+$". "$+\boldsymbol{o}_t$ and replace all the "I" to "PersonX". To build CSKG we use COMET at every step of gameplay as a frozen commonsense generator to produce the tail node $\boldsymbol{v}_j$ given the head node $\boldsymbol{v}_i$ and edge $\boldsymbol{e}_j$ at time step t, formally denoted as $\Pr_\psi(v_{j,t}|\boldsymbol{v}_{j,<t}, \boldsymbol{v}_i, \boldsymbol{e}')$. Figure 1(Right) provides a visualisation of this. COMET takes v_0 as a head node and e_N as an edge and produces v_1 with the corresponding node-to-node score $\phi_{v_0 e_N v_1}$. Multiple tail nodes and node-to-node scores can be generated through the same input and based on the edge, the tail nodes vary dramatically. This process can be applied recursively to the tail nodes, expanding CSKG, i.e. generate tail nodes given v_1 head node with e_N. See Appendix A for more details.

2.2 Commonsense Conditioning

To blend commonsense into the agent's decision, the log-likelihood score is employed to contemplate each component independently. We, then, compute the total score as a weighted sum to promote the natural language action.

Agent-to-Action Score. The score function for the gameplay is obtained from the agent,

$$\phi_{\boldsymbol{a}}^{(k)} = \frac{1}{|\boldsymbol{a}_k|}\sum_{n=1}^{|\boldsymbol{a}_k|}\log \pi_\theta(a_{k,n}|a_{k,<n},\boldsymbol{o}_{t-1}),$$

where $\phi_{\boldsymbol{a}}^{(k)}$ is the agent-to-action score for k action, computed as the sum of log-likelihood of the natural language action. Intuitively, the agent-to-action score signifies how much the action directs to the reward signals. This is learned during the online training of the agent.

Node-to-Action Score. Inspired by Bosselut et al. (2021); Yasunaga et al. (2021), the commonsense level of actions for each generated node is measured using COMET,

$$\phi_{\boldsymbol{v}_i\boldsymbol{e}_j\boldsymbol{a}_k} = \frac{1}{|\boldsymbol{a}_k|}\sum_{n=1}^{|\boldsymbol{a}_k|}\log \Pr_\psi(a_{k,n}|a_{k,<n},\boldsymbol{v}_i,\boldsymbol{e}_j),$$

$$\phi_{\boldsymbol{va}}^{(lk)} = \max_{\boldsymbol{e}}(\phi_{\boldsymbol{v}_l\boldsymbol{e}_1\boldsymbol{a}_k},\phi_{\boldsymbol{v}_l\boldsymbol{e}_2\boldsymbol{a}_k},\cdots),$$

where $\phi_{\boldsymbol{v}_i\boldsymbol{e}_j\boldsymbol{a}_k}$ is the score per $\boldsymbol{va}$ edge, $\boldsymbol{e}\in\mathcal{E}_{\boldsymbol{va}}$, while the node-to-action score is denoted by $\phi_{\boldsymbol{va}}^{(lk)}$ which is the maximum $\phi_{\boldsymbol{v}_i\boldsymbol{e}_j\boldsymbol{a}_k}$ over $\boldsymbol{va}$ edges. The node-to-action score intersects commonsense with action, implying how plausible the action is given the commonsense prediction.

Node-to-Node Score. Additionally, we adopted the score between nodes in CSKG from Bosselut et al. (2021),

$$\phi_{\boldsymbol{v}_i\boldsymbol{e}'_j\boldsymbol{v}_l} = \frac{1}{|\boldsymbol{v}_l|}\sum_{n=1}^{|\boldsymbol{v}_l|}\log\Pr_\psi(v_{l,n}|v_{l,<n},\boldsymbol{v}_i,\boldsymbol{e}'_j),$$

$$\phi_{\boldsymbol{v}}^{(l)} = \max_{\boldsymbol{v},\boldsymbol{e}'}(\phi_{\boldsymbol{v}_1\boldsymbol{e}'_1\boldsymbol{v}_l},\phi_{\boldsymbol{v}_1\boldsymbol{e}'_2\boldsymbol{v}_l},\cdots,\phi_{\boldsymbol{v}_2\boldsymbol{e}'_1\boldsymbol{v}_l},\cdots),$$

where $\phi_{\boldsymbol{v}_i\boldsymbol{e}'_j\boldsymbol{v}_l}$ is the score per head node and $\boldsymbol{vv}$ edges, $\boldsymbol{e}'\in\mathcal{E}_{\boldsymbol{vv}}$, while the node-to-node score is $\phi_{\boldsymbol{v}}^{(l)}$, max of $\phi_{\boldsymbol{v}_i\boldsymbol{e}'_j\boldsymbol{v}_l}$ over head nodes and $\boldsymbol{vv}$ edges.[2] The node-to-node score is designed to promote commonsense triples that are more sensible commonsense-wise.[3]

Total Score. The total score assigned for each action is computed as:

$$\phi = \max_{\boldsymbol{v}}(\gamma_{\boldsymbol{a}}\phi_{\boldsymbol{a}}+\gamma_{\boldsymbol{va}}\phi_{\boldsymbol{va}}+\gamma_{\boldsymbol{v}}\phi_{\boldsymbol{v}}), \tag{1}$$

where ϕ is the total score per action since max is over nodes. The γ coefficients are hyperparameters and balance the weights between different components of the scoring function. Finally, the new conditioned policy is obtained as softmax(ϕ). We refer to this whole process as commonsense conditioning. A visualisation of the overall model is provided in the Figure 1(Left).

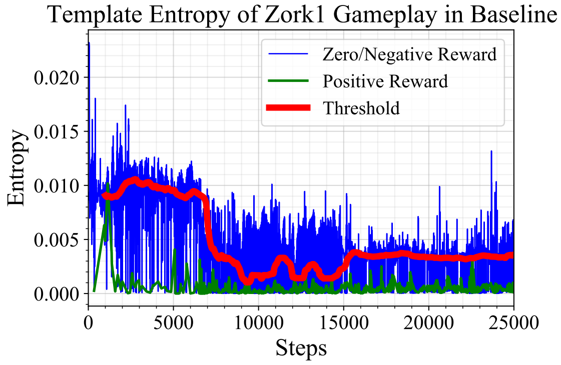


Figure 2: The plot of the entropy of TEMPLATE policy distribution over steps. The green indicates the entropy for a positive reward signal, and the blue does the same for zero or negative rewards. The entropy scheduler threshold of median is plotted as a red curve.

Intuitively, when the agent is not confident in current time-step, the policy distribution is arbitrary, resulting in homogeneous $\phi_{\boldsymbol{a}}$. This would be specifically the case during the initial stage of the training, but can also occur at any stage of the game where the agent cannot predict reward signal in a small number of steps. Under these circumstances, ϕ would be more dictated by $\phi_{\boldsymbol{va}}$ and $\phi_{\boldsymbol{v}}$. Conversely, when the agent is confident, the $\phi_{\boldsymbol{a}}$ for different actions will diverge and ϕ will be directed by both commonsense and the agent.

2.3 Entropy Scheduler

Since our technique uses a large LM for natural language generation, the main drawback with our approach is computational costs. In addition to this, where the agent is confident about acquiring the game score for a given action, commonsense could act as an undesired noise. To reflect on these, we propose the *entropy scheduler* to apply commonsense conditioning based on the confidence, the relative entropy of policy distribution. We collect the last 1000 number of the entropy of the template policy and apply commonsense conditioning if the current entropy is higher than the median. Figure 2 visualizes how the entropy scheduler works during training. This suggests that our entropy scheduler with a median threshold can apply commonsense conditioning to those actions with zero or negative

[2] A set of $\boldsymbol{va}$ edge and $\boldsymbol{v}$ edges can be different, but both are subset of CSKG edge set $\mathcal{E}_{\boldsymbol{vv}},\mathcal{E}_{\boldsymbol{va}}\subseteq\mathcal{E}$.

[3] The example of adequate and poor commonsense phrases are: Given PersonX lost umbrella, PersonX is angry and PersonX is hungry, respectively.

Game	KG-A2C		KG-A2C + COMMEXPL		% Difference	
	Score	PPL	Score	PPL	Score	PPL
balances	9.9	4.96	9.8	3.9	-1.01	**-21.37**
enchanter	19.6	4.47	19.6	3.73	0.0	**-16.56**
library	12.4	5.27	11.5	4.8	-7.26	**-8.92**
ludicorp	16.6	3.81	16.4	3.33	-1.2	**-12.6**
reverb	4.8	4.46	4.5	3.67	-6.25	**-17.71**
spirit	1.8	4.3	2.1	4.18	**16.67**	**-2.79**
zork1	24.7	3.77	30.7	3.49	**24.29**	**-7.43**
zork3	0.069	5.18	0.083	4.13	**20.29**	**-20.27**
ztuu	5.0	5.35	6.9	4.39	**38.0**	**-17.94**
MEAN					**+9.28**	**-13.95**

Table 1: Score and perplexity comparison over 9 game environments, with positive results highlighted by **boldface**. The score is computed as the average over the entire training to signify its performance during the training while perplexity (PPL) is measured for a given root node. The last column denotes the percentage difference between KG-A2C with and without COMMEXPL.

zork1: Kitchen. You are in the kitchen of the white house. A table seems to have been used recently for the preparation of food. A passage leads to the west and a dark staircase can be seen leading upward. A dark chimney leads down and to the east is a small window which is open.			
π	put down glass	open brown	put glass on table
$\hat{\pi}$	put glass on table	put down glass	go up
zork3: It is pitch black. You are likely to be eaten by a grue.			
π	put down lamp	take lamp	turn on lamp
$\hat{\pi}$	turn on lamp	put down lamp	go down

Table 2: An illustrative example of how action selection changes with COMMEXPL. Only top 3 actions are shown for readability. TEMPLATE policy is used for π, i.e. the TEMPLATE probability of put down OBJ is used for put down glass, while $\hat{\pi}$ is the policy conditioned on commonsense.

reward signals. [4]

3 Experiments

We use KG-A2C as our goal-driven baseline agent and compare it with KG-A2C with commonsense in a game suite of Jericho. A set of nine games are selected from Jericho carefully based on genre, including three daily puzzle games (library, ludicorp, reverb) and the rest six fantasy adventure games (balances, enchanter, spirit, zork1, zork3, ztuu). Both game setting and optimal configuration for KG-A2C in Ammanabrolu and Hausknecht (2020) were used in our experiments. *We reduced training steps to* $25,000$ *since our objective is to compare the quality of exploration during the training.* Only hyper-parameters in COMMEXPL have been optimized for fair comparison while all the parameters in COMET were fixed during the training, resulting in the equal trainable parameters regardless of COMMEXPL. Details of the hyper-parameters and the experimental setup can be found in Appendix B.

3.1 Main Results

Similar to Ammanabrolu and Hausknecht (2020), we employed the optimal hyper-parameters fine-tuned on zork1 for nine games in Jericho. Table 1 shows the mean score across the entire training and the perplexity of the action given a root node. The score is to compare whether the agent with commonsense achieves *higher game score during the training*. Doing so implies how fast the agent learns with fewer steps, and therefore, more efficient exploration. Perplexity from LM is used as a metric for the smoothness of natural language action. We used GPT-2 from Huggingface (Wolf et al., 2020).

Score Table 1 shows that with COMMEXPL, the agent tends to acquire the game score more frequent in four gaming environments (spirit, zork1, zork3, ztuu). All four have at least 15% increases in game score during training. However, three environments (balances, enchanter, ludicorp) appear to gain no benefits from using COMMEXPL. On the other hand, the remaining two games (library, reverb) take commonsense negatively, suggesting that the commonsense from COMET acts as a noise with respect to pursuing rewards. Per genre, interestingly, those daily puzzle games are either not influenced or negatively influenced from commonsense inductive bias while four out of six fantasy adventure games benefited from it. We speculate this might be due to the fine-tuning which was also done on a single game, zork1.

Coherency Table 1 shows that commonsense prior reduces perplexity of the natural language actions in all nine games. This is because, unlike the game score that is not directly related to commonsense, the semantic properties of the actions are directly related to commonsense. For environments like balances and reverb, despite the agent taking no benefits from commonsense,

[4] As shown in Appendix C, the training time still remains relatively long due to the natural language generation with a large COMET.

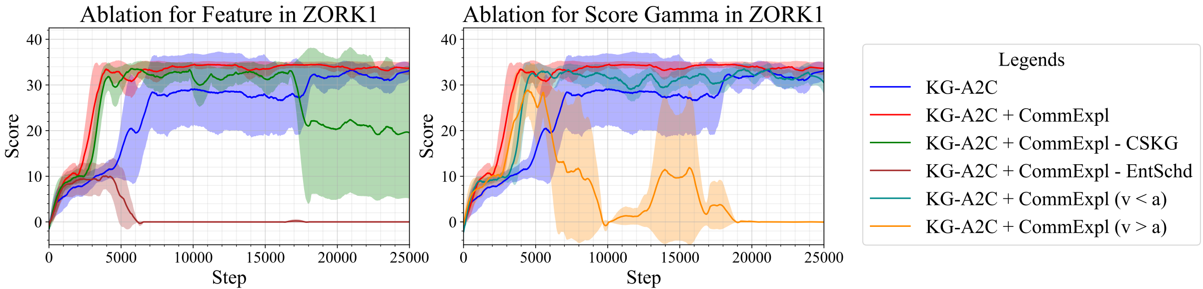


Figure 3: Ablation study on `zork1`. (Left) EntSchd refer to entropy scheduler, so - CSKG and - EntSchd mean we removed CSKG and entropy scheduler from CommExpl. (Right) '>' sign signifies how much commonsense (v) or agent (a) is weighted more over the other.

perplexity drops significantly (*e.g.*, ∼15%). This large reduction in perplexity also appears for fantasy games, in which `zork3` had ∼20% down and `spirit` took as little as ∼3% reduction. This suggests that the game takes advantages on the semantic coherency regardless of whether it helps to achieve high score of the game or the genre of the game.

Qualitative Samples Table 2 provides qualitative samples to show how natural language actions are re-ordered after commonsense conditioning. For instance, in the first example of `zork1`, CommExpl suppresses `open brown` and pushes `put glass on table` to the highest probability. In `zork3`, CommExpl promotes `turn on lamp` over others since the observation informs user that the surrounding is dark.

3.2 Ablation Results

We performed two ablation studies on `zork1` to obtain the optimal hyper-parameters. The first ablation study is for the absence of features, in which we removed CSKG construction and entropy scheduler completely. Thereafter, the changes in score gamma factors have been investigated. The γ coefficients are changed from $(\gamma_{\boldsymbol{v}} = 1, \gamma_{\boldsymbol{va}} = 0.7, \gamma_{\boldsymbol{a}} = 0.8)$ to $(0.4, 0.2, 1)$ for $(\boldsymbol{v} < \boldsymbol{a})$ model and $(1, 1, 0.3)$ for $(\boldsymbol{v} > \boldsymbol{a})$ model.

Feature Figure 3 (Left) shows that the absence of CSKG construction or entropy scheduler causes catastrophic forgetting. KG-A2C is prone to this regardless of commonsense because it does not use any memory component. However, injecting commonsense stochastically enhances the likelihood since the agent follows commonsense when it should not, i.e. a particular action is required to obtain game score. This overlaps with our motivation of entropy scheduler, that *the game score is not directly related to commonsense, so appropriate skipping is necessary*.

Dynamic CSKG contributes to a variety of commonsense, amplifying its commonsense reasoning, and a lack of this will provoke the agent acting more narrow with limited commonsense. Our plot shows that removing CSKG also contributes to the cause of catastrophic forgetting. This suggests that lack of diversity in commonsense may act as a noise to the exploration, and may push the agent to produce more skewed trajectories that cause failure. Therefore, the absence of any component leads to performance decay. Therefore, both are vital components in CommExpl.

Score Gamma Factor The contribution of the commonsense and the agent score is investigated on Figure 3 (Right). By increasing agent's gamma factor, the model acts more alike to the baseline than the optimal hyper-parameters since it trusts its own policy more. Conversely, adding more weights on commonsense leads to catastrophic forgetting. This is caused by the fact that the agent puts too much trust on commonsense, diverging from its own policy excessively. From these, we can conclude that the appropriate balancing is required to make exploration efficient and feasible.

4 Conclusion

We investigated the effect of commonsense in text-based RL agent during the training. Our results show that despite the hyper-parameters tuning on a single game, the proposed approach improves on other gaming environments in Jericho, total four out of nine. Furthermore, injecting commonsense also positively influences the semantics of natural language actions, resulting in lower perplexity. Our future work will extend its application to different text-based environments and investigate how this linguistic properties from LM helps the agent.

References

Ashutosh Adhikari, Xingdi Yuan, Marc-Alexandre Côté, Mikuláŝ Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and William L. Hamilton. 2020. Learning dynamic belief graphs to generalize on text-based games. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Prithviraj Ammanabrolu and Matthew J. Hausknecht. 2020. Graph constrained reinforcement learning for natural language action spaces. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Prithviraj Ammanabrolu and Mark Riedl. 2019. Playing text-adventure games with graph-based deep reinforcement learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3557–3565. Association for Computational Linguistics.

Prithviraj Ammanabrolu, Ethan Tien, Zhaochen Luo, and Mark O. Riedl. 2020. How to avoid being eaten by a grue: Exploration strategies for text-adventure agents. *CoRR*, abs/2002.08795.

Antoine Bosselut, Ronan Le Bras, and Yejin Choi. 2021. Dynamic neuro-symbolic knowledge graph construction for zero-shot commonsense question answering. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 4923–4931. AAAI Press.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4762–4779. Association for Computational Linguistics.

Marc-Alexandre Côté, Ákos Kádár, Xingdi (Eric) Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2018. Textworld: A learning environment for text-based games. In *Computer Games Workshop at ICML/IJCAI 2018*, pages 1–29.

Sahith N. Dambekodi, Spencer Frazier, Prithviraj Ammanabrolu, and Mark O. Riedl. 2020. Playing text-based games with common sense. *CoRR*, abs/2012.02757.

Matthew J. Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. 2020. Interactive fiction games: A colossal adventure. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7903–7910. AAAI Press.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (comet-) atomic 2020: On symbolic and neural commonsense knowledge graphs. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6384–6392. AAAI Press.

Nora Kassner, Philipp Dufter, and Hinrich Schütze. 2021. Multilingual lama: Investigating knowledge in multilingual pretrained language models. In *ACL*, pages 3250–3258.

Zaiqiao Meng, Fangyu Liu, Ehsan Shareghi, Yixuan Su, Charlotte Collins, and Nigel Collier. 2021. Rewire-then-probe: A contrastive recipe for probing biomedical knowledge of pre-trained language models. *arXiv preprint arXiv:2110.08173*.

Farhad Moghimifar, Lizhen Qu, Yue Zhuo, Mahsa Baktashmotlagh, and Gholamreza Haffari. 2020. CosMo: Conditional Seq2Seq-based mixture model for zero-shot commonsense question answering. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5347–5359, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Keerthiram Murugesan, Mattia Atzeni, Pushkar Shukla, Mrinmaya Sachan, Pavan Kapanipathi, and Kartik Talamadupula. 2020. Enhancing text-based reinforcement learning agents with commonsense knowledge. *CoRR*, abs/2005.00811.

Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. 2015. Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, Lisbon, Portugal. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International*

Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 2463–2473. Association for Computational Linguistics.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yunqiu Xu, Meng Fang, Ling Chen, Yali Du, and Chengqi Zhang. 2021. Generalization in text-based games via hierarchical reinforcement learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1343–1353, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yunqiu Xu, Meng Fang, Ling Chen, Yali Du, Joey Tianyi Zhou, and Chengqi Zhang. 2020. Deep reinforcement learning with stacked hierarchical attention for text-based games. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Shunyu Yao, Karthik Narasimhan, and Matthew Hausknecht. 2021. Reading and acting while blindfolded: The need for semantics in text game agents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3097–3102, Online. Association for Computational Linguistics.

Shunyu Yao, Rohan Rao, Matthew Hausknecht, and Karthik Narasimhan. 2020. Keep CALM and explore: Language models for action generation in text-based games. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8736–8754, Online. Association for Computational Linguistics.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

A CSKG Construction

There are three different strategies for building the root node from the textual observation and the natural language action. The most generic one is, given $\boldsymbol{a}_{t-1} =$ “move rug" and $\boldsymbol{o}_t =$ “With a great effort, the rug is moved to one side of the room, revealing the dusty cover of a closed trap door.”, the root node is $\boldsymbol{v}_0 =$ “PersonX ” $+ \boldsymbol{a}_{t-1} +$ “. ” $+ \boldsymbol{o}_t =$“PersonX move rug. With a great effort, the rug is moved to one side of the room, revealing the dusty cover of a closed trap door.”. The example of CSKG with v_0 is in Figure A.1.

However, if the previous action a_{t-1} was not admissible, we set the room description of the textual observation as the root node. Finally, if the action is admissible, but the observation is too short (less than 20 tokens), the root node includes the previous room description of the textual observation at the beginning of the page, $\boldsymbol{v}_0 = \boldsymbol{o}_{\text{room},t-1} +$ “ PersonX ” $+ \boldsymbol{a}_{t-1} +$ “. ” $+ \boldsymbol{o}_t$.

These are motivated from 1) if the previous action is not admissible, the environment is not affected by it, so we simply use the previous room description that captures a lot of information about what the agent can do, 2) if the observation is too short that it does not carry enough information about the situation, we concatenate the previous room description to subjoin the information about surroundings, and 3) otherwise, the generic strategy to build the root node, the previous action and the consequence of it as textual observation.

B Experiment Setup

Action Sampling We set $n_{\texttt{TEMPLATE}}$ to be dynamic, only selecting those based on the probability threshold and validity. The threshold is calculated as 0.75 of its uniform distribution. For instance, zork1 contains 237 number of TEMPLATE, so the threshold is $0.75 \times \frac{1}{237} = 0.00316$. We only select the maximum of 7 TEMPLATE that exceeds the threshold. This avoids a large shift in policy distribution while attaining better computational efficiency. Additionally, we include valid templates to enforce the agent to act more towards on changing the world tree. We sampled objects like KG-A2C since KG-A2C already restricts objects and the actions are usually determined by the template. Therefore, $|\phi_{\boldsymbol{a}}| = n_{\texttt{TEMPLATE}}$, reducing the computations but still covering useful action sets.

Commonsense Transformer Our COMeT is

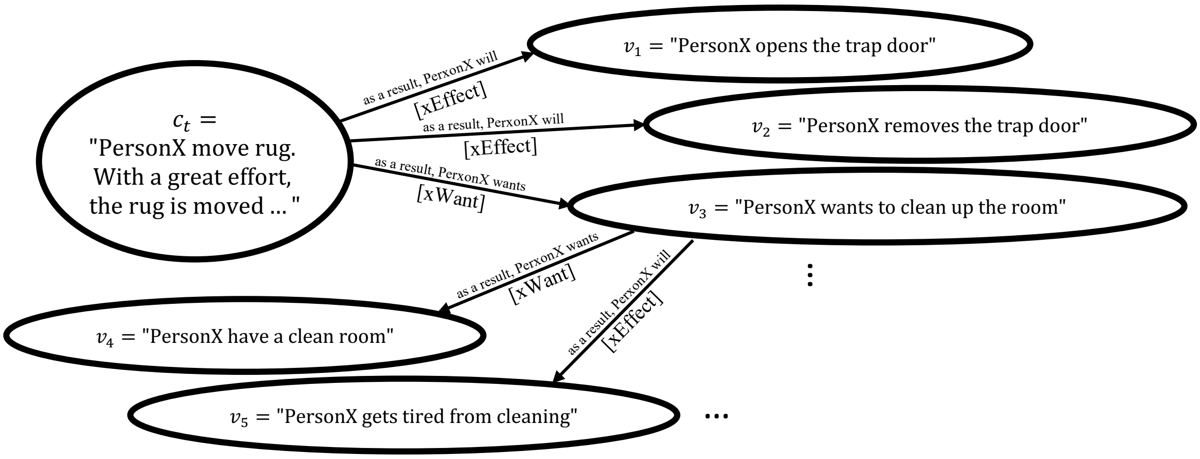

c_t = "PersonX move rug. With a great effort, the rug is moved ... "
as a result, PerxonX will [xEffect]
v_1 = "PersonX opens the trap door"
as a result, PerxonX will [xEffect]
v_2 = "PersonX removes the trap door"
as a result, PerxonX wants [xWant]
v_3 = "PersonX wants to clean up the room"
as a result, PerxonX wants [xWant]
v_4 = "PersonX have a clean room"
as a result, PerxonX will [xEffect]
v_5 = "PersonX gets tired from cleaning"

Figure A.1: The CSKG construction from the corresponding root node with $\boldsymbol{a}_{t-1}$ = "move rug" and $\boldsymbol{o}_t$ = "With a great effort, the rug is moved to one side of the room, revealing the dusty cover of a closed trap door.". Each commonsense phrase node is presented as circle and a directed edge between them is CSKG edge.

BART fine-tuned on ATOMIC-2020 dataset, which is crowdsourced with natural language sentence nodes and 23 commonsense edges (Hwang et al., 2021). We assumed that the general COMET is still good enough to cover TGs. Since the gaming environment runs by the player character, we only focus on the social-interaction commonsense. "xNeed" and "xIntent" are chosen for CSKG construction, $\mathcal{E}_{\boldsymbol{vv}}$, since they deal with what is needed or intended for the event to occur, while "xWant" and "xEffect" for scoring the natural language actions, $\mathcal{E}_{\boldsymbol{va}}$, since they deal with what the player would do following the event. We further set $n_{\text{hop}} = 1$ and $n_{\text{gen}} = 2$ from the observation that they are good enough for zero-shot commonsense question answering (Bosselut et al., 2021; Moghimifar et al., 2020). During the online training of the agent, we freeze the parameters for COMET.

C Computational Expense

The number of node-to-node scores is directly related to the size of CSKG,

$$|\phi_{\boldsymbol{v}}| = \sum_{i=0}^{n_{\text{hop}}} (n_{\text{gen}} \times |\mathcal{E}_{\boldsymbol{vv}}|)^i,$$

where n_{hop} is the number of hops, n_{gen} is the number of triple generation and $\mathcal{E}_{\boldsymbol{vv}}$ is the edge space for CSKG.

On the other hand, the number of node-to-action scores is equal to the number of the total score ϕ,

$$|\phi_{\boldsymbol{va}}| = |\phi| = |\phi_{\boldsymbol{v}}| \times |\mathcal{E}_{\boldsymbol{va}}| \times |\phi_{\boldsymbol{a}}|,$$

where $\mathcal{E}_{\boldsymbol{va}}$ is the edge space for node-to-action score.

We assume $|\phi_{\boldsymbol{a}}| \approx 7$ since we select maximum of 7 templates with highest probability and valid templates. Therefore, in our setting, we can calculate the number of the natural language generations per step per environment as,

$$\begin{aligned}|\phi_{\boldsymbol{v}}| + |\phi_{\boldsymbol{va}}| &= |\phi_{\boldsymbol{v}}| + |\phi_{\boldsymbol{v}}| \times |\mathcal{E}_{\boldsymbol{va}}| \times |\phi_{\boldsymbol{a}}| \\ &= |\phi_{\boldsymbol{v}}| \cdot (1 + |\mathcal{E}_{\boldsymbol{va}}| \times |\phi_{\boldsymbol{a}}|) \\ &\approx \sum_{i=0}^{1} (2 \times 2)^i \cdot (1 + 2 \times 7) \\ &= 75\end{aligned}$$

Finally, we can estimate the average number of natural language generation per step by multiplying the number of environments per step $n_{\text{env}} = 32$ and fraction from entropy scheduler $p \approx 0.5$,

$$\begin{aligned}(|\phi_{\boldsymbol{v}}| + |\phi_{\boldsymbol{va}}|) \times n_{\text{env}} \times p &\approx 75 \times 32 \times 0.5 \\ &= 1200\end{aligned}$$

Throughout the training, we require to perform 1200 natural language generations using a large size COMET per step, so this increases the training time from ×3 upto ×10.

A Simple but Effective Pluggable Entity Lookup Table for Pre-trained Language Models

Deming Ye[1,2], **Yankai Lin**[6], **Peng Li**[6,7], **Maosong Sun**[1,2,3,4,5*], **Zhiyuan Liu**[1,2,3,5]
[1]Dept. of Comp. Sci. & Tech., Institute for AI, Tsinghua University, Beijing, China
[2]Beijing National Research Center for Information Science and Technology
[3]International Innovation Center of Tsinghua University, Shanghai, China
[4]Jiangsu Collaborative Innovation Center for Language Ability, Xuzhou, China
[5]Institute Guo Qiang, Tsinghua University [6]Pattern Recognition Center, WeChat AI
[7]Institute for AI Industry Research (AIR), Tsinghua University
yedeming001@163.com

Abstract

Pre-trained language models (PLMs) cannot well recall rich factual knowledge of entities exhibited in large-scale corpora, especially those rare entities. In this paper, we propose to build a simple but effective **P**luggable **E**ntity **L**ookup **T**able (PELT) on demand by aggregating the entity's output representations of multiple occurrences in the corpora. PELT can be compatibly plugged as inputs to infuse supplemental entity knowledge into PLMs. Compared to previous knowledge-enhanced PLMs, PELT only requires 0.2%∼5% pre-computation with capability of acquiring knowledge from out-of-domain corpora for domain adaptation scenario. The experiments on knowledge-related tasks demonstrate that our method, PELT, can flexibly and effectively transfer entity knowledge from related corpora into PLMs with different architectures. Our code and models are publicly available at https://github.com/thunlp/PELT.

1 Introduction

Recent advance in pre-trained language models (PLMs) has achieved promising improvements in various downstream tasks (Devlin et al., 2019; Liu et al., 2019). Some latest works reveal that PLMs can automatically acquire knowledge from large-scale corpora via self-supervised pre-training and then encode the learned knowledge into their model parameters (Tenney et al., 2019; Petroni et al., 2019; Roberts et al., 2020). However, due to the limited capacity of vocabulary, existing PLMs face the challenge of recalling the factual knowledge from their parameters, especially for those rare entities (Gao et al., 2019a; Wang et al., 2021a).

To improve PLMs' capability of entity understanding, a straightforward solution is to exploit an external entity embedding acquired from the knowledge graph (KG) (Zhang et al., 2019; Liu et al., 2020; Wang et al., 2020), the entity description (Peters et al., 2019), or the corpora (Pörner et al., 2020). In order to make use of the external knowledge, these models usually learn to align the external entity embedding (Bordes et al., 2013; Yamada et al., 2016) to the their original word embedding. However, previous works ignore to explore entity embedding from the PLM itself, which makes their learned embedding mapping is not available in the domain-adaptation. Other recent works attempt to infuse knowledge into PLMs' parameters by extra pre-training, such as learning to build an additional entity vocabulary from the corpora (Yamada et al., 2020; Févry et al., 2020), or adopting entity-related pre-training tasks to intensify the entity representation (Xiong et al., 2020; Sun et al., 2020; Wang et al., 2021b). However, their huge pre-computation increases the cost of extending or updating the customized vocabulary for various downstream tasks.

Model	**#Ent**	**Pre-Comp.**	**D-Adapt**
Zhang et al. (2019)	5.0M	∼160h	No
Wang et al. (2021b)	4.6M	∼3,400h	No
Yamada et al. (2020)	0.5M	∼3,800h	No
PELT (our model)	4.6M	7h	Yes

Table 1: Comparison of recent knowledge-enhanced PLMs. We report the pre-computation of BASE models on Wikipedia entities on a V100 GPU. Pre-Comp.: Pre-computation; D-Adapt: Domain Adaptation.

In this paper, we introduce a simple but effective **P**luggable **E**ntity **L**ookup **T**able (PELT) to infuse knowledge into PLMs. To be specific, we first revisit the connection between PLMs' input features and output representations for masked language modeling. Based on this, given a new corpus, we aggregate the output representations of masked tokens from the entity's occurrences, to recover

[*] Corresponding author: M. Sun (sms@tsinghua.edu.cn)

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 523 - 529
May 22-27, 2022 ©2022 Association for Computational Linguistics

an elaborate entity embedding from a well-trained PLM. Benefiting from the compatibility and flexibility of the constructed embedding, we can directly insert them into the corresponding positions of the input sequence to provide supplemental entity knowledge. As shown in Table 1, our method merely consumes 0.2%∼5% pre-computation compared with previous works, and it also supports the vocabulary from different domains simultaneously.

We conduct experiments on two knowledge-related tasks, including knowledge probe and relation classification, across two domains (Wikipedia and biomedical publication). Experimental results show that PLMs with PELT can consistently and significantly outperform the corresponding vanilla models. In addition, the entity embedding obtained from multiple domains are compatible with the original word embedding and can be applied and transferred swiftly.

2 Methodology

In this section, we first revisit the masked language modeling pre-training objective. After that, we introduce the pluggable entity lookup table and explain how to apply it to incorporate knowledge into PLMs.

2.1 Revisit Masked Language Modeling

PLMs conduct self-supervised pre-training tasks, such as masked language modeling (MLM) (Devlin et al., 2019), to learn the semantic and syntactic knowledge from the large-scale unlabeled corpora (Rogers et al., 2020). MLM can be regarded as a kind of cloze task, which requires the model to predict the missing tokens based on its contextual representation. Formally, given a sequence of tokens $X = (x_1, x_2, \dots, x_n)$, with x_i substituted by [MASK], PLMs, such as BERT, first take tokens' word embedding and position embedding as input and obtain the contextual representation:

$$\boldsymbol{H} = \mathrm{Enc}(\mathrm{LayerNorm}(\mathbf{E}(X) + \boldsymbol{P})), \qquad (1)$$

where $\mathrm{Enc}(\cdot)$ denotes a deep bidirectional Transformer encoder, $\mathrm{LayerNorm}(\cdot)$ denotes layer normalization (Ba et al., 2016), $\mathbf{E} \in \mathbb{R}^{|V| \times D}$ is the word embedding matrix, V is the word vocabulary, P is the absolute position embedding and $\boldsymbol{H} = (\boldsymbol{h}_1, \boldsymbol{h}_2, \dots, \boldsymbol{h}_n)$ is the contextual representation. After that, BERT applies a feed-forward network (FFN) and layer normalization on the contextual representation to compute the output representation of x_i:

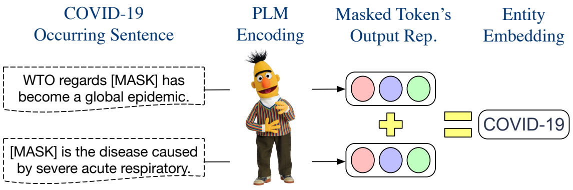

COVID-19 Occurring Sentence
PLM Encoding
Masked Token's Output Rep.
Entity Embedding
WTO regards [MASK] has become a global epidemic.
[MASK] is the disease caused by severe acute respiratory.
COVID-19

Figure 1: An illustration of the our PELT.

$$\boldsymbol{r}_{x_i} = \mathrm{LayerNorm}(\mathrm{FFN}(\boldsymbol{h}_i)). \qquad (2)$$

Since the weights in the softmax layer and word embeddings are tied in BERT, the model calculate the product of $\boldsymbol{r}_{x_i}$ and the input word embedding matrix to further compute x_i's cross-entropy loss among all the words:

$$\begin{aligned}\mathcal{L} &= -\sum \log \Pr(x_i|\boldsymbol{r}_{x_i}) \\ &= -\sum \log \frac{\exp(\mathbf{E}(x_i)^T \boldsymbol{r}_{x_i})}{\sum_{w_j \in V} \exp(\mathbf{E}(w_j)^T \boldsymbol{r}_{x_i})}.\end{aligned} \qquad (3)$$

2.2 Construct Pluggable Entity Embedding

Due to the training efficiency, the vocabulary sizes in existing PLMs typically range from 30K to 60K subword units, and thus PLMs have to disperse the information of massive entities into their subword embeddings. Through revisiting the MLM loss in Eq. 3, we could intuitively observe that the word embedding and the output representation of BERT are located in the same vector space. Hence, we are able to recover the entity embedding from BERT's output representations to infuse their contextualized knowledge to the model.

To be specific, given a general or domain-specific corpus, we design to build the lookup table for entities that occurs in the downstream tasks on demand. For an entity e, such as a Wikidata entity or a proper noun entity, we construct its embedding $\mathbf{E}(e)$ as follows:

Direction A feasible method to add entity e to the vocabulary of PLM is to optimize its embedding $\mathbf{E}(e)$ for the MLM loss with other parameters frozen. We collect the sentences S_e that contain entity e and substitute it with [MASK]. The total influence of $\mathbf{E}(e)$ to the MLM loss in S_e can be formulated as:

$$\begin{aligned}\mathcal{L}(e) &= -\sum_{x_i \in S_e} \log \Pr(e|\boldsymbol{r}_{x_i}) \\ &= \sum_{x_i \in S_e} \log Z_{x_i} - \mathbf{E}(e)^T \sum_{x_i \in S_e} \boldsymbol{r}_{x_i},\end{aligned} \qquad (4)$$

where $Z_{x_i} = \sum_{w_j \in V \cup \{e\}} \exp(\mathbf{E}(w_j)^T \boldsymbol{r}_{x_i})$, x_i is the replaced masked token for entity e and $\boldsymbol{r}_{x_i}$ is the PLM's output representation of x_i.

Compared with the total impact of the entire vocabulary on Z_{x_i}, $\mathbf{E}(e)$ has a much smaller impact. If we ignore the minor effect of $\mathbf{E}(e)$ on Z_{x_i}, the optimal solution of $\mathbf{E}(e)$ for $\mathcal{L}(e)$ is proportional to $\sum_{x_i \in S_e} \boldsymbol{r}_{x_i}$. Hence, we set $\mathbf{E}(e)$ as:

$$\mathbf{E}(e) = C \cdot \sum_{x_i \in S_e} \boldsymbol{r}_{x_i}, \tag{5}$$

where C denotes the scaling factor.

Practically, $\mathbf{E}(e)$ also serves as the negative log-likelihood of other words' MLM loss (Kong et al., 2020). However, Gao et al. (2019a) indicates that the gradient from such negative log-likelihood will push all words to a uniformly negative direction, which weakens the quality of rare words' representation. Here, we ignore this negative term and obtain the informative entity embedding from Eq. 5.

Norm We define $\boldsymbol{p}(e)$ as the position embedding for entity e. Since the layer normalization in Eq. 1 makes the norm $|\mathbf{E}(e) + \boldsymbol{p}(e)|$ to $D^{\frac{1}{2}}$, we find that the norm $|\mathbf{E}(e)|$ has little effect on the input feature of the encoder in use. Therefore, we set the norm of all the entity embeddings as a constant L. Then, we evaluate the model with different L on the unsupervised knowledge probe task and choose the best L for those fine-tuning tasks.

2.3 Infuse Entity Knowledge into PLMs

Since the entity embedding we obtained and the original word embedding are both obtained from the masked language modeling objective, the entity can be regarded as a special input token. To infuse entity knowledge into PLMs, we apply a pair of bracket to enclose the constructed entity embedding and then insert it after the original entity's subwords. For example, the original input,

Most people with COVID-19 have a dry `[MASK]` *they can feel in their chest.*

becomes

*Most people with COVID-19 (**COVID-19**) have a dry* `[MASK]` *they can feel in their chest.*

Here, the entity ***COVID-19*** adopts our constructed entity embedding and other words use their original embedding. We simply convey the modified input to the PLM for encoding without any additional structures or parameters, to help the model predict `[MASK]` as *cough*.

A note on entity links In previous section, we hypothesize that we know the entity linking annotations for the involved string name. In practice, we can obtain the gold entity links provided by some datasets like FewRel 1.0. For the datasets where the linking annotations are not available, we employ a heuristic string matching for entity linking[1].

3 Experiment

3.1 Implementation Details

We choose RoBERTa$_{\texttt{Base}}$ (Liu et al., 2019), a well-optimized PLM, as our baseline model and we equip it with our constructed entity embedding to obtain the PELT model. For the knowledge probe task, we further experiment with another encoder-architecture model, uncased BERT$_{\texttt{Base}}$ (Devlin et al., 2019), and an encoder-decoder-architecture model, BART$_{\texttt{Base}}$ (Lewis et al., 2020).

We adopt Wikipedia and biomedical S2ORC (Lo et al., 2020) as the domain-specific corpora and split them into sentences with NLTK (Xue, 2011). For Wikipedia, we adopt a heuristic entity linking strategy with the help of hyperlink annotations. For the used FewRel 1.0 and Wiki80 datasets, we directly use the annotated linking information. For other datasets, we link the given entity name through a simple string match. For each necessary entity, we first extract up to 256 sentences containing the entity from the corpora. We adopt Wikipedia as the domain-specific corpus for FewRel 1.0, Wiki80 and LAMA, and we adopt S2ORC as the domain-specific corpus for FewRel 2.0. After that, we construct the entity embedding according to Section 2.2.

We search the norm of entity embedding L among 1-10 on the knowledge probe task. We find $L = 7, 10, 3$ performs a bit better for RoBERTa, BERT and BART respectively. In the fine-tuning process, we freeze the constructed embeddings as an lookup table with the corresponding norm. After that, we run all the fine-tuning experiments with 5 different seeds and report the average score.

3.2 Baselines

We select three of the most representative entity-aware baselines, which adopt an external entity embedding, an entity-related pre-training task, or a trainable entity embedding: (1) **ERNIE** (Zhang et al., 2019) involves the entity embedding learned from Wikidata relation (Bordes et al., 2013). We

[1] Details are shown in the Appendix.

Model	Ext. Pretrain	FewRel 1.0				FewRel 2.0			
		5-1	5-5	10-1	10-5	5-1	5-5	10-1	10-5
ERNIE†	✓	$92.7_{\pm0.2}$	$97.9_{\pm0.0}$	$87.7_{\pm0.4}$	$96.1_{\pm0.1}$	$66.4_{\pm1.6}$	$88.2_{\pm0.5}$	$51.2_{\pm0.7}$	$80.1_{\pm1.0}$
KEPLER	✓	$90.8_{\pm0.1}$	$96.9_{\pm0.1}$	$85.1_{\pm0.1}$	$94.2_{\pm0.1}$	$74.0_{\pm1.0}$	$89.2_{\pm0.2}$	$61.7_{\pm0.1}$	$82.1_{\pm0.1}$
LUKE	✓	$91.8_{\pm0.4}$	$97.5_{\pm0.1}$	$85.3_{\pm0.4}$	$95.3_{\pm0.1}$	$64.8_{\pm1.4}$	$89.2_{\pm0.2}$	$46.6_{\pm0.8}$	$80.5_{\pm0.5}$
RoBERTa	-	$90.4_{\pm0.3}$	$96.2_{\pm0.0}$	$84.2_{\pm0.5}$	$93.9_{\pm0.1}$	$71.2_{\pm2.1}$	$89.4_{\pm0.2}$	$53.3_{\pm0.8}$	$83.1_{\pm0.4}$
PELT	-	$\mathbf{92.7}_{\pm0.3}$	$\mathbf{97.5}_{\pm0.0}$	$\mathbf{87.5}_{\pm0.3}$	$\mathbf{95.4}_{\pm0.1}$	$\mathbf{75.0}_{\pm1.3}$	$\mathbf{92.1}_{\pm0.2}$	$\mathbf{60.4}_{\pm1.1}$	$\mathbf{85.6}_{\pm0.2}$

Table 2: The accuracy on the FewRel dataset. N-K indicates the N-way K-shot configuration. Both of FewRel 1.0 and FewRel 2.0 are trained on the Wikipedia domain, and FewRel 2.0 is tested on the biomedical domain. ERNIE† has seen facts in the FewRel 1.0 test set during pre-training. We report standard deviations as subscripts.

Model	1%	10%	100%
ERNIE	$66.4_{\pm0.4}$	$87.7_{\pm0.2}$	$93.4_{\pm0.1}$
KEPLER	$62.3_{\pm1.0}$	$85.4_{\pm0.2}$	$91.7_{\pm0.1}$
LUKE	$63.1_{\pm1.0}$	$86.9_{\pm0.4}$	$92.9_{\pm0.1}$
RoBERTa	$59.8_{\pm1.7}$	$85.7_{\pm0.2}$	$91.7_{\pm0.1}$
PELT	$\mathbf{65.6}_{\pm1.0}$	$\mathbf{88.3}_{\pm0.3}$	$\mathbf{93.4}_{\pm0.1}$

Table 3: The accuracy on the test set of Wiki80. 1%/ 10% indicate using 1%/ 10% supervised training data respectively.

Model	LAMA		LAMA-UHN	
	G-RE	T-REx	G-RE	T-REx
ERNIE	10.0	24.9	5.9	19.4
KEPLER	5.5	23.4	2.5	15.4
LUKE	3.8	32.0	2.0	25.3
RoBERTa	5.4	24.7	2.2	17.0
PELT	**6.4**	**27.5**	**2.8**	**19.3**
BERT	**13.9**	34.9	8.8	26.8
BERT-PELT	13.3	**40.7**	**8.9**	**34.5**
BART	5.1	15.9	1.3	12.0
BART-PELT	**6.9**	**24.4**	**2.1**	**14.9**

Table 4: Mean P@1 on the knowledge probe benchmark. G-RE: Google-RE.

adopt the RoBERTa version of ERNIE provided by Wang et al. (2021b); (2) **KEPLER** (Wang et al., 2021b) encodes textual entity description into entity embedding and learns fact triples and language modeling simultaneously; (3) **LUKE** (Yamada et al., 2020) learns a trainable entity embedding to help the model predict masked tokens and masked entities in the sentences.

3.3 Relation Classification

Relation Classification (RC) aims to predict the relationship between two entities in a given text. We evaluate the models on two scenarios, the few-shot setting and the full-data setting.

The few-shot setting focuses on long-tail relations without sufficient training instances. We evaluate models on FewRel 1.0 (Han et al., 2018) and FewRel 2.0 (Gao et al., 2019b). FewRel 1.0 contains instances with Wikidata facts and FewRel 2.0 involves a biomedical-domain test set to examine the ability of domain adaptation. In the N-way K-shot setting, models are required to categorize the query as one of the existing N relations, each of which contains K supporting samples. We choose the state-of-the-art few-shot framework Proto (Snell et al., 2017) with different PLM encoders for evaluation. For the full-data setting, we evaluate models on the Wiki80, which contains 80 relation types from Wikidata. We also add 1% and 10% settings, meaning using only 1% / 10% data of the training sets.

As shown in Table 2 and Table 3, on FewRel 1.0 and Wiki80 in Wikipedia domain, RoBERTa with PELT beats the RoBERTa model by a large margin (e.g. +3.3% on 10way-1shot), and it even achieves comparable performance with ERNIE, which has access to the knowledge graph. Our model also gains huge improvements on FewRel 2.0 in the biomedical domain (e.g. +7.1% on 10way-1shot), while the entity-aware baselines have little advance in most settings. Compared with most existing entity-aware PLMs which merely obtain domain-specific knowledge in the pre-training phase, our proposed pluggable entity lookup table can dynamically update the models' knowledge from the out-of-domain corpus on demand.

3.4 Knowledge Probe

We conduct experiments on a widely-used knowledge probe dataset, LAMA (Petroni et al., 2019). It applies cloze-style questions to examine PLMs' ability on recalling facts from their parameters. For example, given a question template *Paris is the capital of* `[MASK]`, PLMs are required to predict the masked token properly. In this paper, we not only

Model	[0,10)	[10,50)	[50,100)	[100,+)
RoBERTa	18.1	21.1	25.8	26.1
PELT	**21.9**	**24.8**	**29.0**	**28.7**

Table 5: Mean P@1 on T-Rex with respect to the subject entity's frequency in Wikipedia.

use Gooogle-RE and T-REx (ElSahar et al., 2018) which focus on factual knowledge, but also evaluate models on LAMA-UHN (Pörner et al., 2020) which filters out the easy questionable templates.

As shown in Table 4, without any pre-training, the PELT model can directly absorb the entity knowledge from the extended input sequence to recall more factual knowledge, which demonstrates that the entity embeddings we constructed are compatible with original word embeddings. We also find that our method can also bring huge improvements to both BERT and BART in the knowledge probe task, which proves our method's generalization on different-architecture PLMs.

Effect of Entity Frequency Table 5 shows the P@1 results with respect to the entity frequency. While RoBERTa performs worse on rare entities than frequent entities, PELT brings a substantial improvement on rare entities, i.e., near 3.8 mean P@1 gains on entities that occur less than 50 times.

4 Conclusion

In this paper, we propose PELT, a flexible entity lookup table, to incorporate up-to-date knowledge into PLMs. By constructing entity embeddings on demand, PLMs with PELT can recall rich factual knowledge to help downstream tasks.

Acknowledgement

This work is supported by the National Key R&D Program of China (No. 2020AAA0106502), Institute Guo Qiang at Tsinghua University, and International Innovation Center of Tsinghua University, Shanghai, China. We thank Zhengyan Zhang and other members of THUNLP for their helpful discussion and feedback. Deming Ye conducted the experiments. Deming Ye, Yankai Lin, Xiaojun Xie and Peng Li wrote the paper. Maosong Sun and Zhiyuan Liu provided valuable advices to the research.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

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

Hady ElSahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon S. Hare, Frédérique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4937–4951, Online. Association for Computational Linguistics.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019a. Representation degeneration problem in training natural language generation models. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019b. FewRel 2.0: Towards more challenging few-shot relation classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6250–6255, Hong Kong, China. Association for Computational Linguistics.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel:

A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Lingpeng Kong, Cyprien de Masson d'Autume, Lei Yu, Wang Ling, Zihang Dai, and Dani Yogatama. 2020. A mutual information maximization perspective of language representation learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: enabling language representation with knowledge graph. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 2901–2908. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Nina Pörner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: efficient-yet-effective entity embeddings for BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 803–818. Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5418–5426. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how BERT works. *Trans. Assoc. Comput. Linguistics*, 8:842–866.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4077–4087.

Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. CoLAKE: Contextualized language and knowledge embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3660–3670, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Cunxiang Wang, Pai Liu, and Yue Zhang. 2021a. Can generative pre-trained language models serve as knowledge bases for closed-book qa? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3241–3251. Association for Computational Linguistics.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *CoRR*, abs/2002.01808.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021b. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Trans. Assoc. Comput. Linguistics*, 9:176–194.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Nianwen Xue. 2011. Steven bird, evan klein and edward loper. *Natural Language Processing with Python*. o'reilly media, inc 2009. ISBN: 978-0-596-51649-9. *Nat. Lang. Eng.*, 17(3):419–424.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 250–259. ACL.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

A Heuristic String Matching for Entity Linking

For the Wikipedia, we first create a mapping from the anchor texts with hyperlinks to their referent Wikipedia pages. After that, We employ a heuristic string matching to link other potential entities to their pages.

For preparation, we collect the aliases of the entity from the redirect page of Wikipedia and the relation between entities from the hyperlink. Then, we apply spaCy [2] to recognize the entity name in the text. An entity name in the text may refer to multiple entities of the same alias. We utilize the relation of the linked entity page to maintain an available entity page set for entity disambiguation .

Algorithm 1 Heuristic string matching for entity disambiguation

```
$S \Leftarrow$ { the linked entity page in anchor text}
$E \Leftarrow$ { potential entity name in text}
repeat
    $S' \Leftarrow$ { the neighbor entity pages that have hyperlink or Wikidata relation with pages in $S$}
    $E' \Leftarrow \{e|e \in E$ and $e$ can be uniquely linked to entity page in $S'$ by string matching }
    $E \Leftarrow E - E'$
    $S \Leftarrow E'$
until $S = \phi$
```

Details of the heuristic string matching are shown in Algorithm 1, we match the entity name to surrounding entity page of the current page as close as possible. e will release all the source code and models with the pre-processed Wikipedia dataset.

For other datases, we adopt a simple string matching for entity linking.

B Training Configuration

We train all the models with Adam optimizer (Kingma and Ba, 2015), 10% warming up steps and maximum 128 input tokens. Detailed training hyper-parameters are shown in Table 6.

We run all the experiments with 5 different seeds (42, 43, 44, 45, 46) and report the average score with the standard deviation. In the 1% and 10% settings' experiments for Wiki80, we train the model with 10-25 times epochs as that of the 100% setting's experiment.

For FewRel, we search the batch size among [4,8,32] and search the training step in [1500, 2000, 2500]. We evaluate models every 250 on validation and save the model with best performance for testing. With our hyper-parameter tuning, the results of baselines in FewRel significantly outperforms that reported by KEPLER (Wang et al., 2021b).

Dataset	Epoch	Train Step	BSZ	LR
Wiki80	5	-	32	3e-5
FewRel 1.0	-	1500	32	2e-5
FewRel 2.0	-	1500	32	2e-5

Table 6: Training Hyper-parameters. BSZ: Batch size; LR: Learning rate.

[2] https://spacy.io/

S^4-Tuning: A Simple Cross-lingual Sub-network Tuning Method

Runxin Xu[1*], **Fuli Luo**[2], **Baobao Chang**[1†], **Songfang Huang**[2†], **Fei Huang**[2]
[1]Key Laboratory of Computational Linguistics, Peking University, MOE, China
[2]Alibaba Group
runxinxu@gmail.com, chbb@pku.edu.cn
{lfl259702,songfang.hsf,f.huang}@alibaba-inc.com

Abstract

The emergence of multilingual pre-trained language models makes it possible to adapt to target languages with only few labeled examples. However, vanilla fine-tuning tends to achieve degenerated and unstable results, owing to the *Language Interference* among different languages, and *Parameter Overload* under the few-sample transfer learning scenarios. To address two problems elegantly, we propose S^4-Tuning, a **S**imple Cro**ss**-lingual **S**ub-network Tuning method. S^4-Tuning first detects the most essential sub-network for each target language, and only updates it during fine-tuning. In this way, the language sub-networks lower the scale of trainable parameters, and hence better suit the low-resource scenarios. Meanwhile, the commonality and characteristics across languages are modeled by the overlapping and non-overlapping parts to ease the interference among languages. Simple but effective, S^4-Tuning gains consistent improvements over vanilla fine-tuning on three multilingual tasks involving 37 different languages in total (XNLI, PAWS-X, and Tatoeba).

1 Introduction

Recently, a variety of multilingual pre-trained language models (PLMs) have been proposed, including mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020). Based on these PLMs, it is possible to adapt the model to specific target languages, with only a handful of labeled examples in the downstream tasks, which is called *few-shot cross-lingual transfer learning* (Lauscher et al., 2020; Hedderich et al., 2020; Bari et al., 2021).

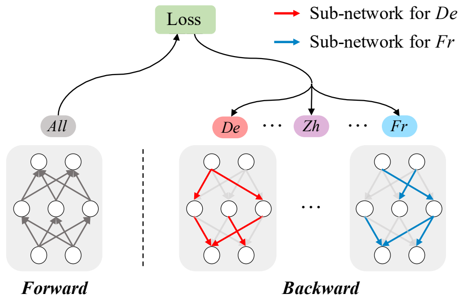


Figure 1: Utilizing full network during the forward process (left), S^4-Tuning only updates a specific sub-network according to the language of the input example (right). Sub-networks are detected based on the importance of model parameters towards different languages.

However, traditional fine-tuning tends to obtain degenerated and unstable results, due to the following two challenges. (1) **Parameter Overload**: Given only few labeled data for a target language, it is challenging to update all model parameters, and such a mismatch between the scale of data and trainable parameters can cause overfitting (Dodge et al., 2020; Zhao et al., 2021). (2) **Language Interference**: Sharing commonality though, different languages also possess their own characteristics. Hence, the adaption towards a specific target language can interfere with that of other languages (Lin et al., 2021), which also damages the transfer performance.

Therefore, it is natural to ask the question, *How to address the Parameter Overload and Language Interference problem **elegantly***? In this paper, we propose a **S**imple Cro**ss**-lingual **S**ub-network Tuning method, S^4-Tuning, which tries to deal with these two problems jointly. As shown in Figure 1, S^4-Tuning detects the most fundamental language sub-networks (with a simple and intuitive criterion in Sec. 3.2), and only updates the specific sub-network corresponding to the input language during training. For one thing, we update the language sub-network on a matching scale, which better suits the low-resource scenarios and addresses the *Parameter Overload* problem. For another, the commonality across languages is modeled by the overlap among different language sub-networks, while the characteristics are also allowed by the

[*]Joint work between Alibaba and Peking University.
[†]Corresponding authors.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 530 - 537
May 22-27, 2022 ©2022 Association for Computational Linguistics

non-overlapping parts. With such a better trade-off, the *Language Interference* problem is alleviated.

Simple to implement, S^4-Tuning also reveals evident effectiveness in the downstream tasks in our experiments. Compared with vanilla fine-tuning, S^4-Tuning consistently offer improvements across different multi-lingual downstream tasks. For example, it improves by 0.9 and 5.6 average points on XNLI and Tatoeba tasks, respectively.

2 Related Work

Towards better few-shot cross-lingual transfer, Zhao et al. (2021) freeze the embedding and encoder layers of the PLM during fine-tuning, which is not effective and flexible enough. Nooralahzadeh et al. (2020) adopt the traditional meta-learning method MAML (Finn et al., 2017), but it is not practical enough, since it requires extra abundant labeled data for meta-training. Differently, we try a more elegant and effective way to handle the *Parameter Overload* and *Language Interference* problem through language sub-networks.

Some works also find a sub-network for each language pair in machine translation (Lin et al., 2021; Xie et al., 2021), or each task in multi-task learning (Sun et al., 2020; Liang et al., 2021). However, their forward and backward are both based on sub-networks, which is more like ***pruning***. Instead, we update parameters within the sub-network during the backward process, but still forward on the whole network to fully utilize the knowledge stored in the entire model. Our work most closely resembles the work of Xu et al. (2021). However, S^4-Tuning deals with multiple sub-networks simultaneously rather than a single sub-network in more challenging few-shot multi-lingual scenarios, and adopts different criteria for language sub-network detection. We empirically show the superiority of S^4-Tuning in Figure 3 in Section 4.5.

3 S^4-Tuning: Simple CrosS-lingual Sub-network Tuning

We formally present the problem formulation (Sec. 3.1). Then we introduce our proposed method, S^4-Tuning, which firstly detects the most important sub-network for each target language (Sec. 3.2), and then only updates the corresponding sub-network during the backward process (Sec. 3.3).

3.1 Problem Formulation

Given a specific task, the original multilingual PLM θ_{pre} is firstly fine-tuned on rich-resource labeled data $\mathcal{D}_s = (\mathcal{X}_s, \mathcal{Y}_s)$ in source language s to obtain θ_s (*source training*) following Lauscher et al. (2020). Then, we aim to better adapt θ_s to multiple target languages $\mathcal{T} = \{t_1, t_2, \ldots, t_{\|\mathcal{T}\|}\}$ with target labeled data $\mathcal{D}_\mathcal{T} = \{(\mathcal{X}_\text{t}, \mathcal{Y}_\text{t}) \mid t \in \mathcal{T}\}$ (*target adapting*). Specifically, suppose there are $\mathcal{C}$ different classes, we have K training examples for each class $c \in \mathcal{C}$ in target language t, and K is remarkably small in low-resource scenarios, leading to $|\mathcal{D}_s| \gg |\mathcal{D}_\mathcal{T}|$. In our paper, we use English as source language following Lauscher et al. (2020).

3.2 Language Sub-network Detection

In this section, we aim to identify the most important sub-network for each target language. In detail, for target language t, if parameter h_i is essential to language t, the change of loss would be large once we remove h_i (i.e., $h_i = 0$) (Molchanov et al., 2017), which is shown in Equation 1 and H refers to other parameters excluding h_i.

$$\Omega^t(h_i) = \left|\mathcal{L}^t(H, h_i = 0) - \mathcal{L}^t(H, h_i)\right| \qquad (1)$$

Following Molchanov et al. (2017), we approximate with Taylor Expansion, and obtain Eq. 2.

$$\Omega^t(h_i) = \left|\frac{\partial \mathcal{L}^t(H, h_i)}{\partial h_i} h_i\right| \qquad (2)$$

Though different scoring criteria can be used, we find this one works best. After deriving the importance score of parameters for target language t based on $(\mathcal{X}_t, \mathcal{Y}_t)$, parameters with the highest score are selected as the sub-network for t. It can be indicated by a mask M_t, where $M^t(h_i) = 1$ if h_i belongs to the sub-network, and $M^t(h_i) = 0$ otherwise. With N parameters in total, we can set up sub-network scale by $p_t = \frac{\sum_{i=1}^{N} M^t(h_i)}{N}$. We unify p_t across different languages as p, that is, $p = p_1 = p_2 = \cdots = p_{\|\mathcal{T}\|}$.

3.3 Constrained Language Adaption

According to the distinctive patterns of language sub-networks, we adapt to the target languages with their most essential parameters.

Forward During the forward procedure, we encode instances by the *full network* regardless of its language. In this way, we can better make full use of the knowledge contained in the whole model.

Method	**ar**	**bg**	**de**	**el**	**en**	**es**	**fr**	**hi**	**ru**	**sw**	**th**	**tr**	**ur**	**vi**	**zh**	**Avg**
K=64																
FC Only	77.43	82.36	82.28	81.51	88.84	83.93	82.48	76.08	79.30	71.55	76.38	78.55	72.37	78.94	78.27	79.35±0.03
FC+Pooler	77.50	82.55	82.44	81.75	88.94	84.20	82.69	76.25	79.75	71.84	76.83	78.96	72.59	79.30	78.64	79.62±0.07
Full Model	78.77	83.73	83.05	81.98	88.32	84.16	83.05	76.67	80.54	72.35	77.42	79.65	73.45	80.10	79.34	80.17±0.53
S^4-Tuning (Ours)	**79.26**	**84.01**	**83.64**	**82.55**	**89.10**	**84.87**	**83.63**	**77.94**	**81.06**	**73.24**	**78.11**	**80.21**	**74.28**	**80.59**	**80.18**	**80.84±0.16**
K=128																
FC Only	77.97	83.01	82.70	81.99	89.04	84.62	82.99	76.63	80.11	72.49	77.13	79.25	73.23	79.54	79.41	80.01±0.02
FC+Pooler	78.06	83.07	82.78	82.10	89.08	**84.66**	**83.15**	76.70	80.17	72.79	77.44	79.44	73.31	79.85	79.46	80.14±0.11
Full Model	78.80	83.61	83.23	82.31	88.43	83.95	82.91	77.01	80.62	72.66	77.65	79.50	73.58	80.29	80.00	80.30±0.28
S^4-Tuning (Ours)	**79.70**	**84.43**	**84.04**	**82.90**	**89.08**	84.61	83.75	**77.93**	**81.38**	**73.67**	**79.03**	**80.47**	**74.64**	**81.24**	**81.13**	**81.20±0.04**

Table 1: **Comparison with other fine-tuning methods on XNLI**. S^4-Tuning consistently outperforms other methods under different K settings, and also achieves lower standard deviation compared with *Full Model* tuning. Although with low standard deviation, *FC Only* and *FC+Pooler* yield inferior results.

Backward Different from vanilla fine-tuning, we only update the parameters within the significant *language sub-network*. It can be achieved by multiplying the gradients with the mask M^t. By this means, we lower the scale of trainable parameters to address *Parameter Overload*, and maintain the commonality and characteristics across different languages to handle *Language Interference*.

4 Experiments

4.1 Datasets

We conduct experiments on three multilingual tasks. Cross-lingual Natural Language Inference (XNLI) (Conneau et al., 2018) is a natural language inference task involving 15 different languages. Besides, Cross-lingual Paraphrase Adversaries from Word Scrambling (PAWS-X) (Yang et al., 2019) focuses on determining whether two sentences are paraphrases with 7 languages. Tatoeba (Artetxe and Schwenk, 2019) with 37 languages is a cross-lingual sentence retrieval task, which finds the nearest neighbor based on cosine similarity between multilingual representations of sentences.

4.2 Experimental Setups

Experiments are based on XLM-R_{large} (Conneau et al., 2020). Following Zhao et al. (2021), we firstly fine-tune the PLM for 10 epochs with batch size 32 on full English labeled examples for *source-training*, whose results are comparable to Hu et al. (2020) (details in Appendix A). Then we continue to fine-tune 5 epochs on K-shot data over target languages, and we use $K \in \{64, 128\}$. The translated examples provided by Hu et al. (2020) are used as the training data for target languages. We search learning rate from {5e-6, 8e-6, 1e-5, 3e-5}, and p from $\{0.1, 0.3, 0.5\}$. We report the average score on the test set of 5 runs with different seeds.

4.3 Main Results

Besides vanilla ***Full Model*** fine-tuning, we also compare with two strong baselines (Zhao et al., 2021): 1) ***FC Only***: Only update the linear classifier during training. 2) ***FC+Pooler***: Only update the linear classifier and pooler layer during training.

S^4-Tuning helps the model better adapt to target languages with strong and stable performance. As shown in Table 1, S^4-Tuning outperforms other fine-tuning methods on XNLI. For example, compared with *Full Model* tuning, S^4-Tuning yields an improvement of up to 0.90 average points, and the standard deviation of multiple random runs is also lowered, suggesting more stable performance. Although with lower standard deviation, *FC Only* and *FC+Pooler* reveal inferior performance. Similar results are observed on PAWS-X task (shown in Appendix B due to limited space), in which S^4-Tuning also beat other methods on both $K = 64$ and $K = 128$ settings, e.g., outperforms *Full Model* tuning by 0.7 average points when $K = 64$.

S^4-Tuning strengthens the model ability to capture cross-lingual semantics, thanks to more precise and flexible adaption for different target languages. We adopt models fine-tuned on PAWS-X through different methods, and search the best encoder layer to derive multilingual sentence representations for Tatoeba task. The most semantically similar sentence is retrieved directly with cosine similarity between representations. As shown in Table 2, S^4-Tuning yields an improvement of up to 5.64 average points across 36 target languages, in comparison with vanilla *Full Model* tuning.

4.4 Similarity Between Sub-networks

In this section, we aim to understand the intrinsic relations among different language sub-networks.

Method	ar	he	vi	id	jv	tl	eu	ml	ta	te	af	nl	de	el	bn	hi	mr	ur	Avg
K=64																			
FC Only/FC+Pooler*	46.7	63.8	73.0	79.2	16.1	36.3	36.4	65.9	26.7	38.5	61.0	82.1	89.0	60.5	43.8	71.5	53.5	25.3	58.5
Full Model	48.8	65.6	76.4	79.8	17.7	38.5	39.5	66.4	31.1	43.5	61	82.6	89.9	61.4	44.4	72.7	55.2	30.8	60.5
S^4-Tuning (Ours)	**55.6**	**69.0**	**81.8**	**82.6**	**20.3**	**44.0**	**46.8**	**71.8**	**43.3**	**55.0**	**67.0**	**84.7**	**92.4**	**66.7**	**52.5**	**76.6**	**59.2**	**49.6**	**66.1**
K=128																			
FC Only/FC+Pooler*	46.7	63.8	73.0	79.2	16.1	36.3	36.4	65.9	26.7	38.5	61.0	82.1	89.0	60.5	43.8	71.5	53.5	25.3	58.5
Full Model	55.5	69.0	82.6	83.6	21.4	42.3	44.9	76.1	38.1	51.9	67.2	85.7	92.6	67.2	51.7	79.6	63.2	43.6	66.2
S^4-Tuning (Ours)	**58.2**	**71.4**	**85.1**	**86.1**	**23.0**	**47.8**	**50.4**	**74.9**	**46.5**	**58.3**	**70.0**	**87.8**	**93.6**	**70.4**	**56.3**	**81.4**	**65.5**	**51.3**	**69.5**

Table 2: **Comparison with other fine-tuning methods on cross-lingual retrieval task Tatoeba** across 36 languages. We only list 18 languages due to limited space, and the complete results are provided in Appendix D. S^4-Tuning consistently achieves the best performance across different target languages. *: Same as the result of the model after source training (θ_s), since these two methods do not update the encoder layers of the model.

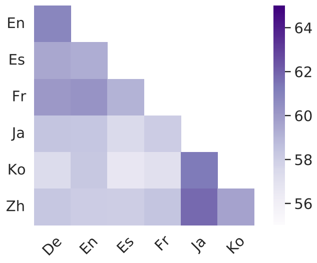
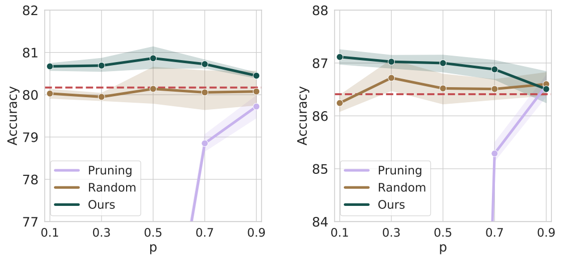


Figure 2: The overlapping ratio between sub-networks of different languages.



(a) XNLI

(b) PAWS-X

Figure 3: Compare S^4-Tuning with **Pruning** and **Random** sub-network across various sub-network ratio p. The red horizontal line denotes the result of vanilla full model tuning. S^4-Tuning reveals superior performance over other strategies.

Specifically, we explore the similarity using the Jaccard similarity coefficient to quantify the overlapping ratio between two sub-networks. Figure 2 illustrates the results based on PAWS-X experiments with $K = 128$ and $p = 0.5$ settings, It can be observed that the eastern languages (*Ja*, *Ko*, *Zh*) are similar to each other, while different from the western languages (*De*, *En*, *Es*, *Fr*). For example, the sub-network of Japanese (*Ja*) is much more similar to that of Korean (*Ko*) and Chinese (*Zh*) than others. It suggests that the detected sub-networks potentially capture the inductive bias of language similarity, and model their commonality and characteristics through overlapping and non-overlapping parts flexibly.

4.5 Comparison with Different Sub-network Strategies: Pruning and Random

To further understand the effect of S^4-Tuning, we compare with two sub-network strategies in XNLI and PAWS-X with $K = 64$: 1) **Pruning** (Lin et al., 2021; Xie et al., 2021): both forward and backward are through a pruned sub-network (while S^4-Tuning uses the full network for forward). We adopt Equation 2 as the criterion to prune the model for all target languages. 2) **Random**: the sub-networks are detected randomly for S^4-Tuning rather than following a specific criterion.

As shown in Figure 3, for pruning, the model would collapse if $p < 0.7$, and the best score achieved in $p = 0.9$ is still lower than the vanilla fine-tuning in XNLI. The performance of random sub-network is slightly lower than vanilla fine-tuning in XNLI, while slightly higher in PAWS-X. Compared with these two strategies, S^4-Tuning achieves the best scores in an overwhelming majority of cases, which suggests the superiority of S^4-Tuning in few-shot cross-lingual transfer.

5 Conclusion

Towards better few-shot cross-lingual transfer learning, we propose S^4-Tuning. S^4-Tuning detects the most essential sub-network for each target language, and only updates these parameters during the backward process, while still utilizing the full model for the forward process. In this way, we reduce the scale of trainable parameters that better suits low-resource scenarios to address overfitting, and better deal with the interference across languages. Our experiments show that S^4-Tuning consistently outperforms other fine-tuning methods in different downstream tasks.

Acknowledgements

This paper is supported by the National Science Foundation of China under Grant No.61936012, the National Key RD Program of China under Grand No.2018AAA0102003, and the National Science Foundation of China under Grant No.61876004.

References

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics (TACL)*.

M. Saiful Bari, Batool Haider, and Saab Mansour. 2021. Nearest neighbour few-shot learning for cross-lingual classification. *arXiv*, arXiv:2109.02221.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv*, arXiv:2002.06305.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*.

Michael A. Hedderich, David Adelani, Dawei Zhu, Jesujoba Alabi, Udia Markus, and Dietrich Klakow. 2020. Transfer learning and distant supervision for multilingual transformer models: A study on African languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Chen Liang, Simiao Zuo, Minshuo Chen, Haoming Jiang, Xiaodong Liu, Pengcheng He, Tuo Zhao, and Weizhu Chen. 2021. Super tickets in pre-trained language models: From model compression to improving generalization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zehui Lin, Liwei Wu, Mingxuan Wang, and Lei Li. 2021. Learning language specific sub-network for multilingual machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistic (ACL)*.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2017. Pruning convolutional neural networks for resource efficient inference. In *International Conference on Learning Representations (ICLR)*.

Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-shot cross-lingual transfer with meta learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Tianxiang Sun, Yunfan Shao, Xiaonan Li, Pengfei Liu, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. Learning sparse sharing architectures for multiple tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Wanying Xie, Yang Feng, Shuhao Gu, and Dong Yu. 2021. Importance-based neuron allocation for multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. 2021. Raise a child in large language model: Towards effective and generalizable fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Mengjie Zhao, Yi Zhu, Ehsan Shareghi, Ivan Vulić, Roi Reichart, Anna Korhonen, and Hinrich Schütze. 2021. A closer look at few-shot crosslingual transfer: The choice of shots matters. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*.

A Results on Source Training

Since our work focuses on the *target adapting*, we ensure the results on *source training* are comparable to others. As shown in Table 3, the obtained results based on our implementation is comparable or even better than those of Hu et al. (2020) in three multi-lingual tasks.

	PAWS-X	XNLI	Tatoeba
Hu et al. (2020)	86.4	79.2	57.3
Ours	86.4	79.6	58.5

Table 3: Align initial results after source training.

B Results on PAWS-X

Table 4 illustrates the results of different fine-tuning methods on PAWS-X task. Compared with vanilla full model tuning, S^4-Tuning achieves better performance with lower standard deviation, which suggests that S^4-Tuning helps the model better adapt to target languages and obtain more stable results.

C Detailed Results on Tatoeba

Table 5 demonstrates the results on the cross-lingual retrieval task, Tatoeba, across 36 different target languages in total. Since *FC Only* and *FC+Pooler* do not update the intermediate encoder layers, their results are both the same as that of the model after source training. It can be observed that S^4-Tuning outperform other methods by 5.6 $\sim$ 7.6 average points under $K = 64$ setting, and 3.2 $\sim$ 11.0 average points under $K = 128$ setting.

D Results on XQuAD

We also explore S^4-Tuning in multilingual question answering task, XQuAD (Artetxe et al., 2020). As shown in Table 6, S^4-Tuning provides improvements on both $K = 64$ and $K = 128$ settings, along with lower standard deviation.

Method	**de**	**en**	**es**	**fr**	**ja**	**ko**	**zh**	**Avg**
K=64								
FC Only	89.07	94.07	90.26	89.70	**80.33**	79.03	82.81	86.47±0.05
FC + Pooler	89.34	93.90	90.05	89.41	80.12	79.42	82.77	86.43±0.07
Full Model	88.80	93.88	89.52	89.35	79.50	**80.78**	**83.04**	86.41±0.70
S^4-Tuning (Ours)	**90.13**	**94.53**	**90.69**	**90.41**	79.96	80.86	83.22	**87.11±0.16**
K=128								
FC Only	89.46	94.37	90.38	89.90	80.73	79.31	82.93	86.73±0.07
FC + Pooler	89.54	94.19	90.29	89.72	80.32	79.67	82.96	86.67±0.06
Full Model	89.19	94.54	90.85	90.43	80.21	80.93	83.23	87.05±0.41
S^4-Tuning (Ours)	**90.19**	**95.01**	**91.13**	**90.75**	**80.85**	**81.71**	**83.56**	**87.60±0.20**

Table 4: **Comparison with other fine-tuning methods on PAWS-X**. S^4-Tuning achieves the best average score across different languages, and also lower the standard deviation compared with *Full Model* tuning.

Method	**ar**	**he**	**vi**	**id**	**jv**	**tl**	**eu**	**ml**	**ta**	**te**	**af**	**nl**	**de**	**el**	**bn**	**hi**	**mr**	**ur**	**fa**
K=64																			
FC Only/FC+Pooler*	46.7	63.8	73.0	79.2	16.1	36.3	36.4	65.9	26.7	38.5	61.0	82.1	89.0	60.5	43.8	71.5	53.5	25.3	71.9
Full Model	48.8	65.6	76.4	79.8	17.7	38.5	39.5	66.4	31.1	43.5	61.0	82.6	89.9	61.4	44.4	72.7	55.2	30.8	73.4
S^4-Tuning (Ours)	**55.6**	**69.0**	**81.8**	**82.6**	**20.3**	**44.0**	**46.8**	**71.8**	**43.3**	**55.0**	**67.0**	**84.7**	**92.4**	**66.7**	**52.5**	**76.6**	**59.2**	**49.6**	**77.7**
K=128																			
FC Only/FC+Pooler*	46.7	63.8	73.0	79.2	16.1	36.3	36.4	65.9	26.7	38.5	61.0	82.1	89.0	60.5	43.8	71.5	53.5	25.3	71.9
Full Model	55.5	69.0	82.6	83.6	21.4	42.3	44.9	**76.1**	38.1	51.9	67.2	85.7	92.6	67.2	51.7	79.6	63.2	43.6	78.8
S^4-Tuning (Ours)	**58.2**	**71.4**	**85.1**	**86.1**	**23.0**	**47.8**	**50.4**	74.9	**46.5**	**58.3**	**70.0**	**87.8**	**93.6**	**70.4**	**56.3**	**81.4**	**65.5**	**51.3**	**80.7**

Method	**fr**	**it**	**pt**	**es**	**bg**	**ru**	**ja**	**ka**	**ko**	**th**	**sw**	**zh**	**kk**	**tr**	**et**	**fi**	**hu**	**Avg**
K=64																		
FC Only/FC+Pooler*	75.9	69.3	83.0	77.4	72.1	74.4	63.5	53.1	60.6	35.0	21.5	68.9	49.6	69.3	52.9	70.3	66.7	58.5
Full Model	77.0	71.6	82.9	79.5	73.0	76.3	65.7	53.8	64.9	39.9	24.0	70.3	48.7	71.8	56.9	74.1	68.7	60.5
S^4-Tuning (Ours)	**79.3**	**73.7**	**83.8**	**82.0**	**76.5**	**80.0**	**74.3**	**56.0**	**69.4**	**59.7**	**25.7**	**76.4**	**53.7**	**75.8**	**62.3**	**80.3**	**75.1**	**66.1**
K=128																		
FC Only/FC+Pooler*	75.9	69.3	83.0	77.4	72.1	74.4	63.5	53.1	60.6	35.0	21.5	68.9	49.6	69.3	52.9	70.3	66.7	58.5
Full Model	80.9	75.0	86.4	83.4	77.3	80.7	73.7	56.9	70.7	54.1	25.2	78.4	54.5	77.6	61.7	79.9	75.2	66.3
S^4-Tuning (Ours)	**83.3**	**77.6**	**87.1**	**85.6**	**81.3**	**83.3**	**76.0**	**63.5**	**73.3**	**61.0**	**28.4**	**80.6**	**58.7**	**80.3**	**66.2**	**82.0**	**76.8**	**69.5**

Table 5: **Detailed results on cross-lingual retrieval task Tatoeba** across 36 languages. S^4-Tuning outperforms vanilla *Full Model* tuning under a overwhelming majority of cases. *: Same as the result of the model after source training (θ_s), since these two methods do not update the encoder layers of the model.

Method	**en**	**es**	**de**	**el**	**ru**	**tr**	**ar**	**vi**	**th**	**zh**	**hi**	**Avg**
K=64												
Full Model	**72.40**	59.14	**60.91**	56.45	**60.30**	**56.27**	53.53	56.79	68.2	**56.22**	57.82	59.82±0.33
S^4-Tuning (Ours)	72.13	**60.30**	60.89	**57.45**	59.87	55.93	**53.92**	**56.92**	**68.44**	55.09	**57.87**	**59.89±0.10**
K=128												
Full Model	72.42	**59.71**	60.34	**57.70**	**60.54**	**56.18**	53.88	57.18	68.40	56.32	58.30	60.09±0.40
S^4-Tuning (Ours)	**72.48**	59.35	**60.54**	57.68	60.47	56.03	**54.13**	**57.98**	**68.79**	**57.24**	**58.62**	**60.30±0.20**

Table 6: **Comparison with Full Model tuning on XQuAD**. S^4-Tuning outperforms Full Model tuning on both $K = 64$ and $K = 128$ settings, with lower standard deviation.

Region-dependent temperature scaling for certainty calibration and application to class-imbalanced token classification

Hillary Dawkins
University of Guelph, Canada
Vector Institute, Toronto, Canada
hdawkins@uoguelph.ca

Isar Nejadgholi
National Research Council Canada
Ottawa, Canada
isar.nejadgholi@nrc-cnrc.gc.ca

Abstract

Certainty calibration is an important goal on the path to interpretability and trustworthy AI. Particularly in the context of human-in-the-loop systems, high-quality low to mid-range certainty estimates are essential. In the presence of a dominant high-certainty class, for instance the non-entity class in NER problems, existing calibration error measures are completely insensitive to potentially large errors in this certainty region of interest. We introduce a region-balanced calibration error metric that weights all certainty regions equally. When low and mid certainty estimates are taken into account, calibration error is typically larger than previously reported. We introduce a simple extension of temperature scaling, requiring no additional computation, that can reduce both traditional and region-balanced notions of calibration error over existing baselines.

1 Introduction

Calibrating the certainty estimates of neural networks is of the utmost importance for interpretability of results and building trust in AI systems. Ideally, if a model outputs some prediction with an associated probability, we would like to interpret that quantity as the probability of a correct prediction (i.e. as a meaningful certainty estimate) (Zadrozny and Elkan, 2001; Niculescu-Mizil and Caruana, 2005). However, contemporary models are consistently over-confident in their output probabilities (Guo et al., 2017).

Guo et al. (2017) demonstrates that over-confident models can arise by overfitting to the Negative Log-Likelihood (NLL) loss, without overfitting to the classification accuracy. Many calibration methods involve modulating the output logits somehow, according to a prescribed functional form. The parameters of the modulation function are learned on the associated *validation* set by minimizing the NLL loss (thereby correcting the overfit). Guo et al. (2017), as well as many subsequent studies (e.g. Müller et al., 2019; Gupta et al., 2021), showcase the surprising effectiveness of temperature scaling, a single-parameter modulation function.

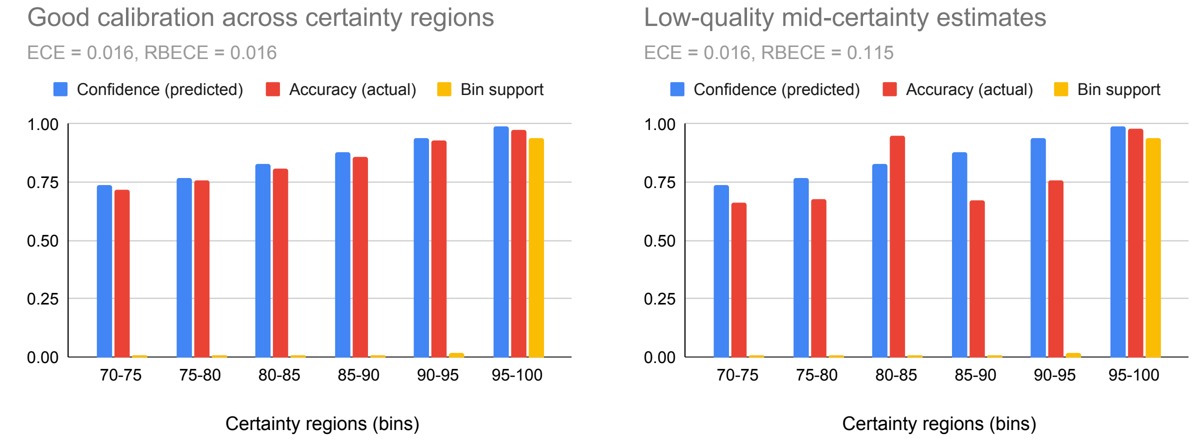
The calibration error is reported as a single quantity computed on the associated test set. Typically, the error is composed of a sum of observed errors across the certainty landscape, visualized using a reliability diagram (DeGroot and Fienberg, 1983; Niculescu-Mizil and Caruana, 2005). However, not all regions contribute equally, especially in the case of class-imbalanced datasets. Consider an output with a predicted certainty of 99.9% vs. an expected actual certainty of 99.8%. In terms of human interpretability and intervention, this difference is negligible. Now consider 79% predicted certainty vs. 71% expected certainty. Clearly the second case is one we should care more about correcting. However, as we will discuss in the following section, the presence of a dominant high-certainty class can cause the first discrepancy to contribute more to the reported calibration error than the second. High quality mid-certainty estimates are most impactful for human-in-the-loop applications, yet current error measures are not sensitive to this region.

Here we take NER (Grishman and Sundheim, 1996; Yadav and Bethard, 2018; Li et al., 2020) as a case study for class-imbalanced token classification. Naturally, the "outside" or non-entity class dominates the dataset. In the following section, we introduce a region-balanced calibration error. We then introduce region-dependent temperature scaling, a calibration method that further reduces error over traditional temperature scaling, across various NER scenarios, without additional computation.

2 Region-balanced expected calibration error

The most popular calibration error metric is the expected calibration error (ECE) (Naeini et al., 2015). A test set is partitioned into certainty bins, each

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 538 - 544
May 22-27, 2022 ©2022 Association for Computational Linguistics



(a) Sample reliability diagram for the case of consistently good certainty estimates across all regions.

(b) Sample reliability diagram for the case of low-quality certainty estimates in the mid-certainty region.

Figure 1: Reliability diagrams contrasting two cases with equal ECE values. Both cases have the same support distribution (yellow), where 90% of all samples have an estimated certainty above 0.95. In each bin, the confidence (blue) is defined as the mean certainty of samples in the bin (i.e. the predicted certainty). The accuracy (red) is the proportion of samples with a correct prediction (i.e. the actual certainty). The calibration error per bin is the difference in predicted and actual certainty. In case (a), calibration error is consistently low across all certainty regions. In case (b), calibration error is high across the mid-certainty regions. However, because of the dominant support in the highest certainty bin, this error is undetected by the ECE measure.

containing samples with a certainty score h within the bin boundaries. The uncalibrated certainty h for a given sample is simply the output probability associated with the predicted class for that sample. Within each bin, we compare the actual and predicted certainty:

$$ECE = \sum_{i} \frac{n_i}{N} |\text{acc}(B_i) - \text{conf}(B_i)| \qquad (1)$$

where $\text{conf}(B_i)$ is the predicted confidence score (the mean h of samples in bin B_i), and $\text{acc}(B_i)$ is the actual accuracy (proportion of correct predictions in bin B_i). Each bin error is weighted by the bin support, where n_i is the number of samples in B_i. If a very high proportion of all samples have a high certainty estimate, only the final bin error has a non-negligible contribution to the overall ECE. Refer to Figure 1 for an illustrated example.

One extension of ECE is to find bin partitions adaptively (Nixon et al., 2020), such that each bin contains an equal number of samples, and each bin contributes equally to the overall error. The result is that many more bins exist in the high certainty region, each of which are narrower in width. Essentially, adaptive-ECE reports the exact same error quantity as ECE in theory, but estimates the quantity using a finer-toothed comb. Neither metric is informative on lower or mid-certainty regions if support is dominated by a high-certainty class.

Maximum expected calibration error (MECE) (Naeini et al., 2015) partially tells the story of low-certainty regions by reporting the maximum bin error. However, MECE is overly sensitive to outlier bins. For example, if a single sample happens to fall in the 0-5% certainty bin, and it has the correct predicted class, we have MECE $> .95$, which is clearly an unusable characterization of the calibration error as a whole.

Here we consider Region-balanced ECE (RB-ECE) as a way to characterize calibration error weighted evenly across certainty regions. Simply,

$$RBECE = \frac{1}{|\Theta|} \sum_{B_i \in \Theta} |\text{acc}(B_i) - \text{conf}(B_i)|. \qquad (2)$$

The error in each bin B_i contributes to the error equally, subject to some threshold support requirement $n_i > \theta$ (to ensure $\text{acc}(B_i)$ is well-defined). The set of bins that meet this requirement is denoted by Θ.

Alternative threshold requirements such as variance in $\text{conf}(B_i)$ vs. bin size could be explored in the future. Another possible extension is custom bin-weighting according to a certainty region of interest for your application (e.g. for human-in-the-loop systems with an intervention criterion).

3 Region-dependent temperature scaling

The idea underlying all calibration methods is generally to modulate overconfident predictions. In traditional temperature scaling (TS), a higher temperature means stronger modulation. Temperature is taken to be a constant, meaning all samples are treated with the same modulation strength.

The idea underlying region-dependent temperature scaling (RD-TS) is simply that the most confident predictions likely need greater modulation than less confident predictions, and therefore temperature should depend on the uncalibrated certainty. If we consider the hypothetical limit of a 0% confidence score, it is intuitive that this does not need any modulation. To investigate this idea empirically, we apply TS to subsets of the OntoNotes dataset, partitioned according to uncalibrated confidence scores. For each confidence region, the ideal temperature is shown in Figure 2. As expected, temperature increases as a function of confidence. A linear fit sufficiently describes the dependence. Within uncertainty, the intercept is equal to the expected value of 1 ($T(h = 0) = 1$, corresponding to no modulation).

To apply RD-TS, uncalibrated logits $\vec{a}$ are scaled as $\vec{q} = \vec{a}/T(h)$ to obtain calibrated logits $\vec{q}$. Temperature is now a function of confidence $T(h) = mh + 1$, where $h = \max(\text{softmax}(\vec{a}))$ is the probability estimate for the predicted class on each sample. The slope m is the single parameter controlling modulation strength.

To estimate m, one could repeat temperature scaling on multiple data subsets, collect data points, and fit the slope as in Figure 2. However, this method increases computational overhead. Instead, let us estimate m from the original TS constant T_0 and some knowledge of the validation dataset which was used to compute T_0. Each sample in the validation set has an ideal temperature, here taken to be in the form $T_i = mh_i + 1$. Assuming each sample contributed to the found T_0 equally, $T_0 = \frac{1}{N}\sum_i^N (mh_i + 1)$. Given access to the validation set, this sum can be computed exactly to find m. However, we can further approximate the sum by loosely assuming that the data has a high proportion of samples (say $\approx 90\%$) with very high certainty estimates (say $\approx .99$ on average). Then the sum is dominated by the first leading term, $T_0 \approx .9(.99m + 1)$. This quick sketch is sufficient to achieve good error reduction over the baseline TS method. The numerical exactness is not too important, but rather the general signature of a high proportion of high-certainty samples is sufficient. We take this further approximation to gain the advantage that nothing specifically needs to be known about the calibration dataset. I.e. If a large pre-trained model has been calibrated on a large or private dataset, and the corresponding temperature T_0 is known, RD-TS can be applied to your model outputs without access to the calibration data or further computation.

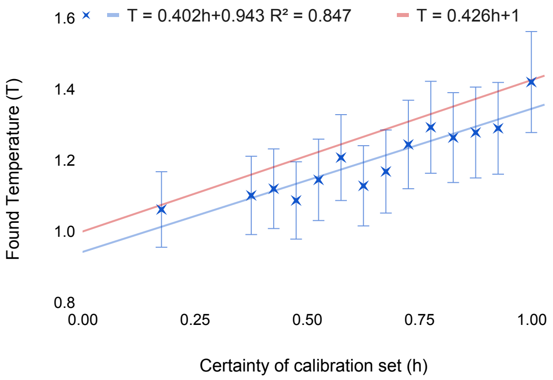


Figure 2: The OntoNotes 5.0 validation set is split into 14 bins according to uncalibrated confidence scores h. For each subset, regular temperature scaling is applied to find the ideal T_0 as a function of average confidence. Blue: Linear regression fit of empirical data ($m = .402 \pm .108$, $b = .943 \pm .073$ with a 95% confidence interval). Red: Region-dependent temperature scaling parameter $T(h)$ as determined by our protocol (see points 1-3). Both methods produce equivalent results within the uncertainty.

In summary, the RD-TS method is performed as follows:

1. Perform regular temperature scaling to obtain T_0, or obtain a previously published T_0 for your model.

2. Find the linear dependence parameter $m = (T_0 - .9)/.89$.

3. Apply calibration to logits $\vec{a}$ as $\vec{q} = \vec{a}/T(h)$, $T = mh + 1$.

RD-TS is a simple extension of temperature scaling which requires no additional training. Like temperature scaling, RD-TS cannot change the predicted class or model accuracy (unlike some other generalizations, vector and matrix scaling).

Scenario	**Uncal.**	**TS**	**VS**	**MS**	**WTS**	**RD-TS**
Classic	.09328	.02543 ($T_0 = 1.28$)	.07040	.06940	.05236	**.02151** ($m = .426$)
Rare & emerging	.09878	.05777 ($T_0 = 1.39$)	.07490	.04932	.11559	**.03549** ($m = .550$)
Fine-grained	.05333	.02179 ($T_0 = 1.12$)	.03440	.04628	.03278	**.01263** ($m = .243$)
Specialized	.07088	.04147 ($T_0 = 1.29$)	.03844	.03590	.03820	**.02781** ($m = .439$)
Sparse training	.09683	.07820 ($T_0 = 1.10$)	.11653	.09528	.06279	**.04110** ($m = .229$)
Differing sources	.05730	.05960 ($T_0 = 1.09$)	.10824	.08470	.05551	**.04019** ($m = .214$)

Table 1: Region-balanced expected calibration error (RBECE); refer to eq. 2.

Scenario	**Uncal.**	**TS**	**VS**	**MS**	**WTS**	**RD-TS**
Classic	.02001	.00862 ($T_0 = 1.28$)	.01359	.01083	.00962	**.00155** ($m = .426$)
Rare & emerging	.04278	.02323 ($T_0 = 1.39$)	.02585	.01580	.04712	**.00949** ($m = .550$)
Fine-grained	.02287	**.00783** ($T_0 = 1.12$)	.01587	.01786	.01462	.00839 ($m = .243$)
Specialized	.01555	.00617 ($T_0 = 1.29$)	.00608	**.00573**	.00631	.00651 ($m = .439$)
Sparse training	.03267	.02190 ($T_0 = 1.10$)	.03113	.02599	**.01645**	.01798 ($m = .229$)
Differing sources	.00950	.00723 ($T_0 = 1.09$)	.01211	.01344	.01020	**.00383** ($m = .214$)

Table 2: Expected calibration error (ECE); refer to eq. 1.

Dataset	$h\|(P = .9)$	$P\|(h = .99)$
OntoNotes	.998	.964
W-NUT 17	.997	.953
Few-nerd	.972	.801
BC2GM	.997	.968
OntoNotes (tc)	.999	.978

Table 3: The mean certainty h of the top .9 most certain samples, $h|(P = .9)$, and the proportion of samples we need to take such that the mean certainty is .99, $P|(h = .99)$. All datasets refer to the corresponding validation set, which is used for calibration. As shown, all datasets have the general signature of a high proportion of high-certainty samples, yet the exact numerical values can deviate from our sketch.

4 Experimental results

4.1 Baseline methods

As RD-TS is a simple extension of regular temperature scaling, we focus comparison on similar post-training parametric calibration methods:

Temperature scaling (TS): Uncalibrated logits $\vec{a}$ are scaled by a single constant T_0 (as $\vec{q} = \vec{a}/T_0$) before softmax is applied to obtain calibrated probability estimates over all classes (Guo et al., 2017).

Vector (generalized Platt) scaling (VS): A generalization of TS such that logits are scaled by $2k$ learned parameters, $\vec{q} = \vec{v} \circ \vec{a} + \vec{b}$, where k is the number of classes (Platt, 1999; Niculescu-Mizil and Caruana, 2005; Guo et al., 2017).

Matrix scaling (MS): A further generalized linear transformation such that logits are scaled by $k^2 + k$ learned parameters, $\vec{q} = M\vec{a} + \vec{b}$ (Guo et al., 2017).

Weighted temperature scaling (WTS): TS using a class-weighted NLL loss during convergence (Obadinma et al., 2021).

4.2 Datasets

We take the NER task as a case study. Datasets represent several important scenarios in token classification settings more broadly:

Classic: The OntoNotes 5.0 NER dataset (Weischedel et al., 2013) represents a baseline “classic” scenario involving plentiful training and calibration data from robust sources.

Rare and emerging named entities: The W-NUT NER dataset[1] (Derczynski et al., 2017) is gathered from noisy social media data which contains difficult entities (e.g. “kktny") due to informal and evolving language.

Fine-grained and few-shot: Few-nerd[2] (Ding et al., 2021) is a challenging few-shot NER dataset with 66 fine-grained entity types (e.g. “art-film").

Specialized language: The BioCreative II Gene Mention Recognition (BC2GM) dataset[3] (Smith et al., 2008) is composed of scientific text where named entities are gene mentions.

[1] huggingface.co/datasets/wnut_17

[2] huggingface.co/datasets/dfki-nlp/few-nerd

[3] huggingface.co/datasets/bc2gm_corpus

Sparse training data: OntoNotes telephone call data is used for training while the full OntoNotes dataset is used for calibration and evaluation. The telephone call data subset is a sparse representation since it is very heavily skewed to the non-entity outside class, and entity mentions are concentrated on “person" and “location", compared to the full OntoNotes dataset (generally containing much richer entity mentions from news sources).

Differing language sources: OntoNotes broadcast news data is used for training, and telephone call data is used for calibration and evaluation. Broadcast news language is professional and grammatically correct. Telephone call language is casual, fragmented and incoherent at times.

4.3 Implementation notes

All NER models use DistilBERT[4] (Sanh et al., 2019) as the base pre-trained model, fine-tuned for NER using the train dataset for each scenario as described above. Further details and performance on the NER task are provided in Appendix A.

Calibration is performed using the uncalibrated logits of the associated validation set as model inputs. Calibration parameters are learned by minimizing the NLL (or weighted NLL) loss for 50 epochs (using SGD with 0.01 learning rate, and 0.9 momentum). Calibration error is computed on the associated test set. To compute both ECE (eq. 1) and RBECE (eq. 2), the number of bins is set to 20. To compute RBECE, the threshold for support per bin is set to $\theta = 40$. The code needed to reproduce these results is made publicly available[5]. All datasets are publicly available with preset train/validation/test data splits.

4.4 Results

Experimental results are summarized in Tables 1 and 2. When low and mid-certainty regions are taken into account by the RBECE, calibration error is larger than previously thought (as reported by ECE). In all scenarios, RD-TS produces the smallest RBECE (in many cases quite substantially). Additionally, RD-TS improves the traditional ECE in the majority of scenarios. The results show that RD-TS is an effective extension of TS across a range of temperature (T_0) values.

Recall in Section 3, we sketch a way to estimate the modulation parameter m, and this approximation follows from assuming that a high proportion of all samples in the calibration set (say $\approx .9$) have a high certainty estimate (say $\approx .99$ on average). We claim that the numerical exactness of these values is not too important (and therefore RD-TS outperforms TS across a range of datasets). This claim is supported empirically (Table 3).

5 Discussion and Conclusion

Good quality mid-range certainty estimates are essential for productive human-model interactions. Despite this, existing calibration error measures can be insensitive to all but the highest certainty regions. We propose a region-balanced error metric to probe this unreported information. When low and mid-certainty regions are taken into account, greater calibration errors are revealed.

Further, we explore the idea of a certainty-dependent temperature. While previous generalizations of TS, such as vector and matrix scaling, allow certainty dependence by increasing the number of learned parameters, these methods are generally outperformed by TS (Guo et al., 2017). Rather than allowing a complicated certainty dependence, we enforce a simple linear dependence (motivated by intuition and an empirical example) without introducing any learnable parameters. Unlike vector and matrix scaling, RD-TS cannot change the relative ranking of logits, and therefore model accuracy is retained (in single-label settings). One line of future work could be to apply RD-TS on top of weighted temperature scaling, a method known to decrease variance in calibration error among classes (Obadinma et al., 2021). Another line of work would be to investigate whether improved certainty estimates can increase model accuracy (in multi-label settings where predictions are applied by meeting a certainty threshold), especially in out-of-domain problems.

Finally, it is important to note that our discussion of a region-balanced error measure, as well as our sketch derivation of the RD-TS method, have been generally applicable to **any problem with a dominant proportion of high-certainty predictions**. This situation does arise in any token classification problem with a dominant “easy” class, as is the case in NER, however this situation can equally occur in class-balanced situations. Therefore, region-dependent temperature scaling can find utility beyond NER, token classification, or class-imbalanced situations.

[4] huggingface.co/transformers/model_doc/distilbert.html
[5] github.com/hillary-dawkins/RegDepTempScaling

Ethical Considerations

We proposed a novel method to calibrate class-imbalanced token classifiers, and demonstrated the method for NER models. This calibration method is a step toward responsible use of AI by offering a measure of reliability, but also has risks that should be considered from an ethical point of view. Calibrated scores are a measure of transparency, and users can interpret a well-calibrated model better. However, all transparency methods expose AI systems to malicious attacks by providing more information about the internal workings of the system. This risk should be taken into account in sensitive tasks, e.g. when an NER model is used to extract personally identifiable information for privacy reasons. Also, users should be warned that a low calibration error does not guarantee robustness in out-of-domain settings. Therefore, in the case of safety-critical tasks such as medical applications of NER, a low calibration error should be interpreted with caution.

Further, low calibration errors should not be used to justify inherently unethical tasks or those out of the scope of the capabilities of NLP technologies. Every task should be evaluated in terms of feasibility and ethical use regardless of reliability and transparency of trained models. It is also important to keep in mind that a well-calibrated model can become miscalibrated as the data changes, and continuous calibration is needed to deal with the ever-changing nature of language.

References

Morris H. DeGroot and Stephen E. Fienberg. 1983. The comparison and evaluation of forecasters. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 32(1/2):12–22.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-NERD: A few-shot named entity recognition dataset. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3198–3213, Online. Association for Computational Linguistics.

Ralph Grishman and Beth Sundheim. 1996. Message Understanding Conference- 6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1321–1330. JMLR.org.

Kartik Gupta, Amir Rahimi, Thalaiyasingam Ajanthan, Thomas Mensink, Cristian Sminchisescu, and Richard Hartley. 2021. Calibration of neural networks using splines. In *International Conference on Learning Representations*.

J. Li, A. Sun, J. Han, and C. Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge & Data Engineering*, (01):1–1.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. 2015. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 2901–2907. AAAI Press.

Alexandru Niculescu-Mizil and Rich Caruana. 2005. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, page 625–632, New York, NY, USA. Association for Computing Machinery.

Jeremy Nixon, Mike Dusenberry, Ghassen Jerfel, Timothy Nguyen, Jeremiah Liu, Linchuan Zhang, and Dustin Tran. 2020. Measuring calibration in deep learning.

Stephen Obadinma, Hongyu Guo, and Xiaodan Zhu. 2021. Class-wise calibration: A case study on covid-19 hate speech. *Proceedings of the Canadian Conference on Artificial Intelligence*. Https://caiac.pubpub.org/pub/vd3v9vby.

John C. Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Larry Smith, Lorraine K. Tanabe, Rie Johnson nee Ando, Cheng-Ju Kuo, I.-Fang Chung, Chun-Nan Hsu, Yu-Shi Lin, Roman Klinger, Christoph M. Friedrich, Kuzman Ganchev, Manabu Torii, Hongfang Liu, Barry Haddow, Craig A. Struble, Richard J. Povinelli, Andreas Vlachos, William A. Baumgartner, Lawrence Hunter, Bob Carpenter, Richard Tzong-Han Tsai, Hong-Jie Dai, Feng Liu, Yifei Chen, Chengjie Sun, Sophia Katrenko, Pieter Adriaans, Christian Blaschke, Rafael Torres, Mariana Neves, Preslav Nakov, Anna Divoli, Manuel Maña-López, Jacinto Mata, and W. John Wilbur. 2008. Overview of BioCreative II gene mention recognition. *Genome Biol.*, 9(2):1–19.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. OntoNotes Release 5.0.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Bianca Zadrozny and Charles Elkan. 2001. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 609–616, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

A NER performance

NER models were obtained by fine-tuning DistilBERT, using the default configuration, for 3 epochs (with learning rate of 2e-5, and weight decay of 0.01). The performance of all NER models is provided in Table A.1 for reference.

Dataset	**P**	**R**	**F**	**A**
OntoNotes	.778	.621	.691	.976
W-NUT 17	.543	.234	.327	.938
Few-nerd	.639	.679	.659	.906
BC2GM	.802	.844	.822	.965
OntoNotes (bc)	.711	.753	.732	.973

Table A.1: For all datasets that were used to train an NER model, we report the precision (P), recall (R), F-score (F) and accuracy (A) of the model on the corresponding test set.

Developmental Negation Processing in Transformer Language Models

Antonio Laverghetta Jr. and **John Licato**
Advancing Machine and Human Reasoning (AMHR) Lab
Department of Computer Science and Engineering
University of South Florida
Tampa, FL, USA
{alaverghett,licato}@usf.edu

Abstract

Reasoning using negation is known to be difficult for transformer-based language models. While previous studies have used the tools of psycholinguistics to probe a transformer's ability to reason over negation, none have focused on the types of negation studied in developmental psychology. We explore how well transformers can process such categories of negation, by framing the problem as a natural language inference (NLI) task. We curate a set of diagnostic questions for our target categories from popular NLI datasets and evaluate how well a suite of models reason over them. We find that models perform consistently better only on certain categories, suggesting clear distinctions in how they are processed.[1]

1 Introduction

Negation is an important construct in language for reasoning over the truth of propositions (Heinemann, 2015), garnering interest from philosophy (Horn, 1989), psycholinguistics (Zwaan, 2012), and natural language processing (NLP) (Morante and Blanco, 2020). While transformer language models (TLMs) (Vaswani et al., 2017) have achieved impressive performance across many NLP tasks, a great deal of recent work has found that they do not process negation well, and often make predictions that would be trivially false in the eyes of a human (Rogers et al., 2020; Ettinger, 2020; Laverghetta Jr. et al., 2021).

In developmental psychology, there has likewise been a great deal of interest in how a child's ability to comprehend negation emerges in the early years of life (Nordmeyer and Frank, 2013, 2018b; Reuter et al., 2018; Grigoroglou et al., 2019). Unlike in NLP, which typically treats negation as representing a single monolithic competency, this research has long understood that there are many kinds of negation used in everyday interactions (Bloom, 1970; Pea, 1982). This ranges from using negation to express a child's rejection of something to clarifying a child's knowledge. These "developmental" categories of negation do not emerge simultaneously; children tend to start using certain kinds before others (Nordmeyer and Frank, 2018a).

Given that these categories represent some of the earliest uses of negation among humans, understanding how well TLMs can master them is important for building more human-like models of language processing. Understanding how well models perform on different categories will indicate whether they have mastery of some forms of negation, while also helping to identify failure points. Another interesting question is whether the proficiency of TLMs on these categories is at all related to competencies in human children (e.g., is the category which models consistently perform the best on the same that children most frequently employ?). However, to our knowledge, no prior work in NLP has focused on how well models perform on the forms of negation of interest to developmental psychology.

In this short paper, we investigate how well a suite of TLMs can process developmental negation,[2] by framing the problem as a natural language inference (NLI) task. We develop a rule-based parser to extract problems from existing NLI datasets, and evaluate our models on each category, in order to determine *(i)* whether certain categories are more solvable by our models than others, and *(ii)* what relationships exist among the categories. We find that models can consistently achieve stronger performance only on certain categories, and that training on combinations or sequences of these categories does not substantially improve a model's downstream performance.

[1] Code and data to reproduce our experiments can be found on Github: https://github.com/Advancing-Machine-Human-Reasoning-Lab/negation-processing-ACL-2022

[2] By which we mean the forms of negation studied in development psychology.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 545 - 551
May 22-27, 2022 ©2022 Association for Computational Linguistics

2 Related Work

Negation is known to be frequently used in everyday conversation. While this includes its logical form, we primarily focus on negation's psycholinguistic forms, especially those that have been studied in the context of developmental psychology. Negation emerges early in child development, with 'no' sometimes being a child's first word (Schneider et al., 2015), and even infants appear to understand forms of negation (Piaget, 1980; Hochmann and Toro, 2021). Preschool children use at least three different kinds of negation (Bloom, 1970), but possibly as many as nine (Choi, 1988). As noted by Nordmeyer and Frank (2018a), one of the first categories children use is *rejection*, where a child rejects an object or activity. This is later followed by *existence*, where a child might express the lack of an object, and later still *denial*, which a child uses to deny the truth of a claim. Larger scale studies of child-directed speech have found that truth-functional kinds of negation tend to emerge later (Liu and Jasbi, 2021), but individual children do vary in their specific order of acquisition (Nordmeyer and Frank, 2018a). It is unknown whether this ordering reflects any deeper dependencies among the different categories, or whether the ordering is reflected in how artificial language models (LMs) learn negation.

In NLP, methods from psycholinguistics have been used to probe the reasoning capabilities of LMs. Results from some studies have indicated that TLMs are not human-like in their processing of negation (Ettinger, 2020; Kassner and Schütze, 2020). A similar line of work has used the NLI task to probe a model's ability to process negation and found that TLMs will often alter their predictions when negation is inserted or removed, even when the negation does not alter the entailment relationship (Hossain et al., 2020; Hartmann et al., 2021). As argued by Kruszewski et al. (2016), part of the challenge of modeling purely logical negation is that a predicate often occurs in very similar contexts regardless of whether it is being negated. They argue that we should view negation as being a "graded similarity function", and show that distributional models can predict human plausibility judgments quite well, even in the presence of negation. These works show that it is unclear how well distributional models, especially TLMs, are actually processing negation. We contribute to this literature from a new perspective, by studying how well models can reason over forms of negation common in developmental psychology.

Category	# Train	# Test
Possession (PO)	1053	520
Existence (EX)	5528	2723
Labeling (L)	2241	1104
Prohibition (PR)	814	400
Inability (I)	1384	682
Epistemic (EP)	1903	936
Rejection (R)	1737	856

Table 1: Summary statistics for the curated dataset.

3 The Developmental Negation Corpus

We use the NLI task to study the negation reasoning capabilities of our models. NLI problems consist of two sentences: a premise (p) and hypothesis (h), and solving such a problem involves assessing whether p textually entails h. The generic structure of the NLI task makes it suitable for studying a variety of underlying reasoning skills, including negation. We specifically use the SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018) datasets.

To automatically identify questions that contain a specific kind of negation, we rely on the work by Liu and Jasbi (2021) which studied how frequently different kinds of developmental negation occur in child-directed speech, using the data from the CHILDES corpus (MacWhinney, 2014). To do this, they created a simple rule-based parser to automatically tag each sentence in CHILDES with the type of negation it contained (if any). We reimplement their parser, in some cases tweaking the rules slightly to better suit the structure of the NLI task. For each example across all the splits of both datasets, we first obtain a dependency parse of both p and h using the diaparser package (Wang et al., 2019), and check if either contains an explicit negation marker ("no", "not", or "n't"). If one span contains negation, we check if the syntactic structure obeys the rules of any of our categories. If the span falls into a category, we mark it as belonging to that category. We use these questions as the diagnostic set for our experiments, splitting out 1/3 of the questions in each category as a *diagnostic test* set, and leaving the remainder as a *diagnostic train* set (and we will refer to them as such). We place the remaining NLI questions containing no negation in a separate NLI_{train} set, giving us about 730,000 examples we use to finetune our models on the NLI task. We split out 9,000 questions from this train set at random to use as a NLI_{dev} set, bal-

Category	Premise	Hypothesis
PO	yeah you probably don't have the right temperatures...	You probably have ideal temperatures...
EX	This analysis pooled estimates...	The analysis proves that there is no link...
L	Not orders, no.	It is not orders.
PR	Two people are sitting against a building near shopping carts.	Run that way but don't run into the...
I	His manner was unfortunate, I observed thoughtfully.	I could not pick out what kind of manner he...
EP	yeah i don't know why	I know why
R	I lowered my voice...	I didn't want to be overheard.

Table 2: NLI examples extracted from each category, long examples have been trimmed to fit on one line.

anced for each label. In the following, we describe the precise rules used to determine which category a negated example should be assigned to:

Possession (PO) We require that the lemma of the root be *have*, *has*, or *had*, and that the root is directly modified by both the negation and the verb *do*.

Existence (EX) We require that *there* occur in the text and precede the negative marker and that the negative marker directly modifies a noun phrase, determiner, or an adverb.

Labeling (L) We require that the sentence begin with either *That* or *It*, and that the root of the sentence is a noun which is modified by *is* or *'s*.

Prohibition (PR) We require that the sentence not contain a subject and that the negation is immediately preceded by *do*. To not conflate this category with others, we filter out cases where the root contains one of the explicit markers of another category (e.g., *like* or *want* in the case of rejection).

Inability (I) We require that the negation directly modify the root of the sentence, and that the word immediately before the negation is either *can* or *could* (e.g., *can not do*). Prior literature has typically viewed inability from an egocentric perspective. However, we found that allowing only the first person severely restricted the number of examples extracted, and therefore chose to also allow the second and third person.

Epistemic (EP) We require that the root be *remember*, *know*, or *think*, and that the root be directly modified by the verb *do*.

Rejection (R) We require that the lemma of the root word be either *like* or *want*, and that the root is modified by the negative marker.

After performing extraction, categories L and PR contained fewer than 1000 examples, which we deemed was insufficient to split into separate train and test sets. To address this, we developed a simple data augmentation approach that utilized the Wordnet database (Miller, 1998). From the dependency parse of both p and h, we check if the root of either parse occurs in both spans. If it does, we obtain all synonyms of the word in Wordnet and replace the root in both spans with the synonym (doing this for every synonym). We found this simple approach increased the number of examples for both L and PR to at least 1500. Note that we performed no augmentation for the other categories, as our parser extracted at least 1500 examples for all other cases. Table 1 shows statistics for the dataset after augmentation.

Table 2 shows extracted examples, along with their category assignment. We generally found that the extracted examples matched up with the prototypical category quite well, although in some cases their semantics differed slightly. For instance, consider a PR example with p = *don't miss having a flick through the albums* and h = *The pictures of old Madeira show a more interesting city than now*, which is an MNLI example originally extracted from a travel guide. Although this technically counts as PR, it does not have quite the same semantics as an actual command. Unfortunately, these ambiguities are not easily resolved, given that negation takes on many forms and may occur at any location within a sentence. We, therefore, opted to focus on forms of negation that can be easily extracted, and leave improvements to our dataset creation protocol for future work.

4 Experiments

Using the curated dataset, we performed a series of exploratory experiments to help us understand how well TLMs process each of the negation categories. We use BERT (Devlin et al., 2019), and RoBERTa (Liu et al., 2019), two popular transformer LMs that have demonstrated impressive results on a variety of language understanding tasks. We also examine MiniBERTa (Warstadt et al., 2020) and BabyBERTa (Huebner et al., 2021), which are both based on the RoBERTa architecture but were pretrained on a much smaller number of tokens (10 million and 5 million respectively), which is more realistic to the amount of language a child is exposed to in the first few years of life. We use the Huggingface implementation of all models (Wolf et al., 2020), and use both the *base* and *large* version of BERT and RoBERTa, which differ only in the number of trainable parameters.

Experiment 1: We began by investigating whether TLMs would master certain negation categories sooner than others over the course of training. We train our models on NLI_{train} for 10 epochs, using a learning rate of $1e-5$, a weight decay of 0.01, a batch size of 16, and a maximum sequence length 175.[3] We selected these hyperparameters to be similar to those which were previously reported to yield strong results when training on NLI datasets (Laverghetta Jr. et al., 2021). We additionally evaluated the models on NLI_{dev}, and found that they all achieved a Matthews Correlation of at least 0.6 (Matthews, 1975), and thus concluded that these hyperparameters were suitable. For every end of epoch checkpoint across all models, we obtained evaluation results on each diagnostic test set. Importantly, the models are not finetuned on any negated NLI questions for this experiment, meaning that all knowledge of negation comes from pre-training. Results are shown in Figure 1. We see that the categories have similar rankings in terms of accuracy. For example, L and PO are among the top two best-performing categories, while R is generally one of the worst-performing ones, indicating clear distinctions in how LMs process the categories. BabyBERTa, unlike other models, also shows stronger similarities to how children acquire negation. For instance, while R is thought to be one of the first categories children acquire, BabyBERTa is the only model where R is one of the highest-ranking categories in terms of accuracy.

Experiment 2: One might expect that children develop a more abstract understanding of negation as they are exposed to different categories. This was suggested by Pea (1978) who argued that more abstract forms of negation develop from less abstract ones, suggesting that mastering one form of negation can lead to positive transfer on others. In Experiment 2, we examined how much positive

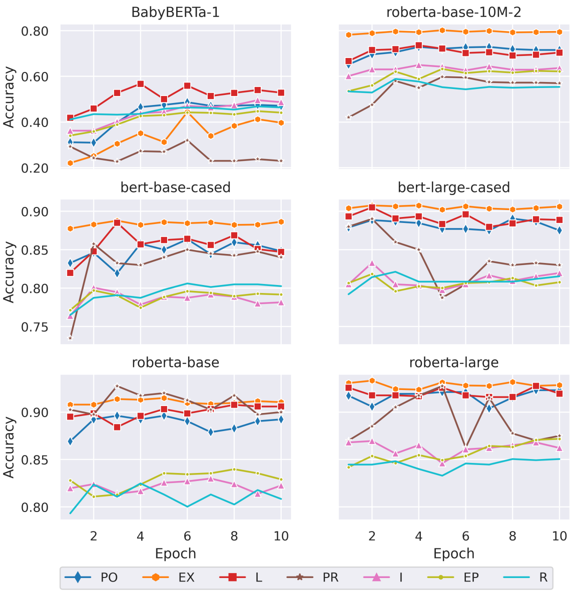


Figure 1: Performance of models finetuned on NLI_{train} for each diagnostic test set. We refer to MiniBERTa using its Huggingface model ID (*roberta-base-10M-2*).

transfer could be obtained from training on one of the negation categories, and then testing on the others. We adopt a similar methodology to Pruksachatkun et al. (2020), who explored the conditions that affect intermediate task transfer learning. Using the models trained in Experiment 1, we further finetune these models for 25 epochs on each diagnostic train set separately. We then evaluate the finetuned models on each diagnostic test set, which allows us to examine all possible pairwise interactions among categories. Figure 2 shows the results for all combinations of diagnostic categories for training and testing. Surprisingly, we find that positive transfer generally only occurs when a model is trained on the same category it is being tested on. Training on a different category has little to no effect on the target category. BabyBERTa is again an exception, as we do see positive transfer for most pairs, suggesting the model is generalizing across categories

Experiment 3: Building on Experiment 2, we examined how the performance of our models is affected when trained on all diagnostic categories in sequence. Assuming that no positive transfer exists among the categories, we would expect to see a model's performance on a particular category improve only after it has been trained on that same category, and even training on multiple other categories should not substantially improve perfor-

[3] We set the maximum sequence length for BabyBERTa to 128, which is the longest that the model supports.

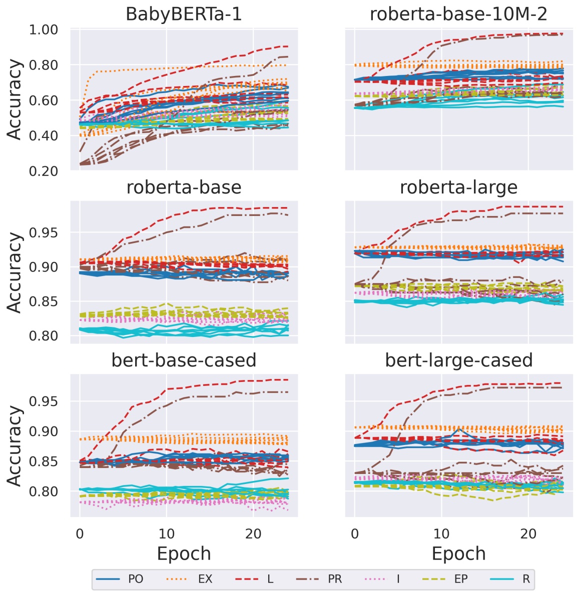


Figure 2: Accuracy of each model on every diagnostic test set, after being finetuned on every diagnostic train set. Plots are color-coded based on the target category.

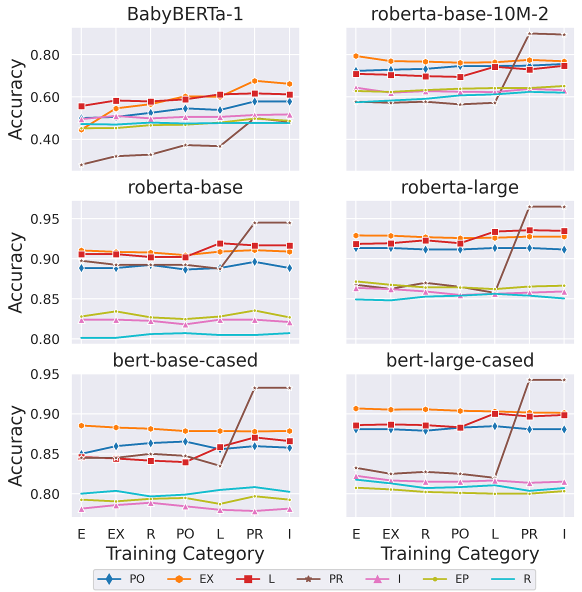


Figure 3: Results from Experiment 3. The x-axis shows the sequence of categories on which all models were trained, while the y-axis shows the accuracy obtained after being trained on a category.

mance on the target. Using the models from Experiment 1, we finetune each model for 10 epochs on every diagnostic train set, using the sequence of categories shown in the x-axis of Figure 3. Additionally, we under-sample all diagnostic train sets to have the same number of questions as PR, so that all categories contribute the same amount of data. Figure 3 shows the results. For some categories, such as L and PR, we see the expected trend. The largest accuracy gain for these categories occurs whenever the model is trained on the same category it is being tested on, and performance drops slightly after being trained on others. However, for categories such as R, the best performance gain is not always after being trained on the same category. We sometimes see the model continue to improve on R after being trained on R, and in some cases, training on R causes performance on R to *decrease*.

5 Discussion and Conclusion

In this paper, we have explored how well transformers process categories of developmental negation. We find that performance rankings across categories are generally consistent, but that the categories seem to test for orthogonal skills in the majority of LMs. In BabyBERTa, we see significant similarities with the order of negation acquisition in children. Two of the best performing categories are R and L, while two of the worst are EX and PR, which aligns quite well to the order observed by Liu and Jasbi (2021). It thus seems that TLMs do at least partially reflect the order of negation acquisition observed in children, although more experiments would be needed to understand the extent of this correlation. That we found category rankings to generally be consistent across LMs may have interesting implications, and understanding why LMs struggle with certain categories may help to improve the ability of LMs to process negation.

Future work can build on these experiments in several ways. In Experiments 2 and 3, we modeled interactions among the negation categories in either a pairwise or sequential fashion, which is unlikely to reflect how children are exposed to negation. More experiments, mixing all of the categories at once in various proportions, might yield a more realistic model of cognitive development. Our approach also requires that each category fits into a specific structure, which limits the amount of examples that can be extracted. Future work will need to expand our ruleset to include more variations in the negated utterances covered. Finally, while we primarily focus on finetuning, pre-training is likely to impact the proficiency of our models on the categories as well. Future work should precisely control the prevalence of each category in the pretraining corpus, to observe what effect this has on downstream performance.

References

Lois Bloom. 1970. Language development: Form and function in emerging grammars. mit research monograph, no 59.

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

Soonja Choi. 1988. The semantic development of negation: a cross-linguistic longitudinal study. *Journal of child language*, 15(3):517–531.

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

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Myrto Grigoroglou, Sharon Chan, and Patricia A Ganea. 2019. Toddlers' understanding and use of verbal negation in inferential reasoning search tasks. *Journal of experimental child psychology*, 183:222–241.

Mareike Hartmann, Miryam de Lhoneux, Daniel Hershcovich, Yova Kementchedjhieva, Lukas Nielsen, Chen Qiu, and Anders Søgaard. 2021. A multilingual benchmark for probing negation-awareness with minimal pairs. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 244–257, Online. Association for Computational Linguistics.

F. H. Heinemann. 2015. VIII.—The Meaning of Negation. *Proceedings of the Aristotelian Society*, 44(1):127–152.

Jean-Rémy Hochmann and Juan M. Toro. 2021. Negative mental representations in infancy. *Cognition*, 213:104599. Special Issue in Honour of Jacques Mehler, Cognition's founding editor.

Laurence Horn. 1989. *A Natural History of Negation*. University of Chicago Press.

Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. 2020. An analysis of natural language inference benchmarks through the lens of negation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9106–9118, Online. Association for Computational Linguistics.

Philip A. Huebner, Elior Sulem, Fisher Cynthia, and Dan Roth. 2021. BabyBERTa: Learning More Grammar With Small-Scale Child-Directed Language. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 624–646, Online. Association for Computational Linguistics.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Germán Kruszewski, Denis Paperno, Raffaella Bernardi, and Marco Baroni. 2016. There is no logical negation here, but there are alternatives: Modeling conversational negation with distributional semantics. *Computational Linguistics*, 42(4):637–660.

Antonio Laverghetta Jr., Animesh Nighojkar, Jamshidbek Mirzakhalov, and John Licato. 2021. Can transformer language models predict psychometric properties? In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 12–25, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zoey Liu and Masoud Jasbi. 2021. English negative constructions and communicative functions in child language. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 43.

Brian MacWhinney. 2014. *The CHILDES project: Tools for analyzing talk, Volume II: The database*. Psychology Press.

Brian W Matthews. 1975. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Roser Morante and Eduardo Blanco. 2020. Recent advances in processing negation. *Natural Language Engineering*, pages 1–10.

Ann Nordmeyer and Michael Frank. 2013. Measuring the comprehension of negation in 2-to 4-year-old children. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 35.

Ann Nordmeyer and Michael C Frank. 2018a. Individual variation in children's early production of negation. In *CogSci*.

Ann E Nordmeyer and Michael C Frank. 2018b. Early understanding of pragmatic principles in children's judgments of negative sentences. *Language Learning and Development*, 14(4):262–278.

Roy D Pea. 1978. *The development of negation in early child language*. Ph.D. thesis, University of Oxford.

Roy D Pea. 1982. Origins of verbal logic: Spontaneous denials by two-and three-year olds. *Journal of child language*, 9(3):597–626.

Jean Piaget. 1980. *Experiments in Contradiction*. University of Chicago Press.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247, Online. Association for Computational Linguistics.

Tracy Reuter, Roman Feiman, and Jesse Snedeker. 2018. Getting to no: Pragmatic and semantic factors in two-and three-year-olds' understanding of negation. *Child development*, 89(4):e364–e381.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Rose M Schneider, Daniel Yurovsky, and Mike Frank. 2015. Large-scale investigations of variability in children's first words. In *CogSci*, pages 2110–2115. Citeseer.

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

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019. Second-order semantic dependency parsing with end-to-end neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4618, Florence, Italy. Association for Computational Linguistics.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020. Learning Which Features Matter: RoBERTa Acquires a Preference for Linguistic Generalizations (Eventually). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Rolf A Zwaan. 2012. The experiential view of language comprehension: How is negation represented. *Higher level language processes in the brain: Inference and comprehension processes*, page 255.

Canary Extraction in Natural Language Understanding Models

Rahil Parikh
Institute for Systems Research
University of Maryland

Christophe Dupuy
Amazon Alexa AI

Rahul Gupta
Amazon Alexa AI

Abstract

Natural Language Understanding (NLU) models can be trained on sensitive information such as phone numbers, zip-codes etc. Recent literature has focused on Model Inversion Attacks (ModIvA) that can extract training data from model parameters. In this work, we present a version of such an attack by extracting canaries inserted in NLU training data. In the attack, an adversary with open-box access to the model reconstructs the canaries contained in the model's training set. We evaluate our approach by performing text completion on canaries and demonstrate that by using the prefix (non-sensitive) tokens of the canary, we can generate the full canary. As an example, our attack is able to reconstruct a four digit code in the training dataset of the NLU model with a probability of 0.5 in its best configuration. As countermeasures, we identify several defense mechanisms that, when combined, effectively eliminate the risk of ModIvA in our experiments.

1 Introduction

Natural Language Understanding (NLU) models are used for different tasks such as question-answering (Hirschman and Gaizauskas, 2001), machine translation (Macherey et al., 2001) and text summarization (Tas and Kiyani, 2007). These models are often trained on crowd-sourced data that may contain sensitive information such as phone numbers, contact names and street addresses. Nasr et al. (2019), Shokri et al. (2017) and Carlini et al. (2018) have presented various attacks to demonstrate that neural-networks can leak private information. We focus on one such class of attacks, called Model Inversion Attack (ModIvA) (Fredrikson et al., 2015), where an adversary aims to reconstruct a subset of the data on which the machine-learning model under attack is trained on. We also demonstrate that established ML practices (e.g. dropout) offer strong defense against ModIvA.

In this work, we start with inserting potentially sensitive target utterances called 'canaries'[1] along with their corresponding output labels into the training data. We use this augmented dataset to train an NLU model $\boldsymbol{f_\theta}$. We perform a open-box attack on this model, i.e., we assume that the adversary has access to all the parameters of the model, including the word vocabulary and the corresponding embedding vectors. The attack takes the form of text completion, where the adversary provides the start of a canary sentence (e.g., 'my pin code is') and tries to reconstruct the remaining, private tokens of an inserted canary (e.g., a sequence of 4 digit tokens). A successful attack on $\boldsymbol{f_\theta}$ reconstructs all the tokens of an inserted canary. We refer to such a ModIvA as 'Canary Extraction Attack' (CEA). In such an attack, this token reconstruction is cast as an optimization problem where we minimize the loss function of the model $\boldsymbol{f_\theta}$ with respect to its inputs (the canary utterance), keeping the model parameters fixed.

Previous ModIvAs were conducted on computer vision tasks where there exists a continuous mapping between input images and their corresponding embeddings. However, in the case of NLU, the discrete mapping of tokens to embeddings makes the token reconstruction from continuous increments in the embedding space challenging. We thus formulate a discrete optimization attack, in which the unknown tokens are eventually represented by a one-hot like vector of the vocabulary length. The token in the vocabulary with the highest softmax activation is expected to be the unknown token of the canary. We demonstrate that in our attack's best configuration, for canaries of type *"my pin code is* $k_1k_2k_3k_4$", $k_i \in \{0, 1, \ldots, 9\}, 1 \leq i \leq 4$, we are able to extract the numeric pin $k_1k_2k_3k_4$ with an accuracy of 0.5 (a lower bound on this accuracy using a naive random guessing strategy for a combination of four digits equals 1×10^{-4}).

[1] Following the terminology in Carlini et al. (2018)

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 552 - 560
May 22-27, 2022 ©2022 Association for Computational Linguistics

Since we present a new application of ModIvA to NLU models, defenses against them are an important ethical consideration to prevent harm and are explored in Section 6. We observe that standard training practices commonly used to regularize NLU models successfully thwart this attack.

2 Related Work

Significant research has been conducted in the field of privacy-preserving machine learning. Shokri et al. (2017) determine whether a particular datapoint belongs to the training set $\boldsymbol{X_{tr}}$. The success of such attacks has prompted research in investigating them (Truex et al., 2019; Hayes et al., 2017; Song and Shmatikov, 2019). Carlini et al. (2018) propose the quantification of unintended memorization in deep networks and presents an extraction algorithm for data that is memorized by generative models. Memorization is further exploited in Carlini et al. (2020) where instances in the training data of very large language-models are extracted by sampling the model. The attacks described above are closed-box in nature where the adversary does not cast the attack as an optimization problem but instead queries the model multiple times.

Open-box ModIvA were initially demonstrated on a linear-regression model (Fredrikson et al., 2014) for inferring medical information. It has been extended to computer vision tasks such as facial recognition (Fredrikson et al., 2015) or image classification (Basu et al., 2019). Our work is a first attempt at performing ModIvAs on NLP tasks.

3 Attack Setup

We consider an NLU model $\boldsymbol{f_\theta}$ that takes an utterance $\boldsymbol{x}$ as input and uses the word-embeddings $\boldsymbol{E(x)}$ for the tokens in $\boldsymbol{x}$ to perform a joint intent classification (IC) and named-entity recognition (NER) task. We assume an adversary with open-box access to $\boldsymbol{f_\theta}$, which means that they are aware of the model architecture, trained parameters $\boldsymbol{\theta}$, loss function $\boldsymbol{L(f_\theta(E(x)), y)}$, label set $\boldsymbol{Y}$ of intents and entities supported by the model and vocabulary $\boldsymbol{V}$ which is obtained from the word-embeddings matrix $\boldsymbol{W} \in \mathbb{R}^{|V| \times d}$. However, the adversary does not have access to the training data $\boldsymbol{X_{tr}}$ used to train $\boldsymbol{f_\theta}$. The adversary's goal is to reconstruct a (private) subset $\boldsymbol{\hat{x}} \subseteq \boldsymbol{X_{tr}}$.

To perform a CEA on $\boldsymbol{f_\theta}$, we keep the parameters $\boldsymbol{\theta}$ fixed and minimize the loss function $\boldsymbol{L}$ with respect to the unknown inputs (i.e., tokens) of a given utterance. This is analogous to a traditional learning problem, except with fixed model parameters and a learnable input space. In this work, we use the NLU model architecture described in Section 4.1.

3.1 Canary Extraction Attacks

We consider a canary sentence $\boldsymbol{x_c} = \boldsymbol{(x_p, x_u)}$, $\boldsymbol{x_c} \in \boldsymbol{X_{tr}}$ with tokens $(p_1, .., p_m, u_1.., u_n)$ and output label $\boldsymbol{y_c} \in \boldsymbol{Y}$. The first m tokens in $\boldsymbol{x_c}$ represent a known prefix $\boldsymbol{x_p}$ (e.g."my pin code is") and the next n tokens $(u_1, .., u_n)$ represent the unknown tokens that an attacker is interested in reconstructing $\boldsymbol{x_u}$ (e.g."one two three four").
We represent the set of word embeddings of this canary $\boldsymbol{E(x_c)}$ as $\boldsymbol{(e_{p_1}, .., e_{p_m}, e^{'}_{u_1}, .., e^{'}_{u_n})}$.

A trivial attack to identify the n unknown tokens in $\boldsymbol{x_u}$ is by directly optimizing $\boldsymbol{L(f_\theta(E(x_c)), y_c)}$ over $\boldsymbol{(e^{'}_{u_1}, .., e^{'}_{u_n})}$, where $\boldsymbol{(e^{'}_{u_1}, .., e^{'}_{u_n})}$ are randomly initialized. Words corresponding to the optimized values of $\boldsymbol{(e^{'}_{u_1}, .., e^{'}_{u_n})}$ are then assigned by identifying the closest vectors in the embedding matrix $\boldsymbol{W}$ using a distance metric (e.g. Euclidean distance). However, our experiments demonstrate that this strategy is not successful since the updates are performed in a non-discrete fashion, whereas the model $\boldsymbol{f_\theta}$ has a discrete input space. We thus focus on performing a discrete optimization, inspired by works on relaxing categorical variables to facilitate efficient gradient flow (Jang et al., 2016; Song and Raghunathan, 2020), as illustrated in Figure 1.

We define a logit vector $\boldsymbol{z_i} \in \mathbb{R}^{|V|}$ for each token $u_i \in \boldsymbol{x_u}$. We then apply a softmax activation with temperature T to obtain $\boldsymbol{a_i} \in \mathbb{R}^{|V|}$:

$$a_{i,v} = \frac{e^{\frac{z_{i,v}}{T}}}{\sum_{j=1}^{|V|} e^{\frac{z_{j,v}}{T}}} \quad \text{for v =1, 2, \ldots,} |V| \qquad (1)$$

$\boldsymbol{a_i}$ is a differentiable approximation of the arg-max over the logit vector for low values of T. This vector then selectively attends to the tokens in the embedding matrix, $\boldsymbol{W} \in \mathbb{R}^{|V| \times d}$, resulting in the embeddings $\boldsymbol{(e^{'}_{u_1}, .., e^{'}_{u_n})}$ used as inputs fed to the model during the attack:

$$\boldsymbol{e^{'}_{u_i}} = \boldsymbol{W^T} \cdot \boldsymbol{a_i} \quad \text{for } 1 \leq i \leq n \qquad (2)$$

We then train our attack and optimize for $\boldsymbol{Z} \in \mathbb{R}^{n \times |V|}$, with $\boldsymbol{Z} = (\boldsymbol{z_1}, \ldots, \boldsymbol{z_n})$:

$$\boldsymbol{\hat{Z}} = \arg\min_{\boldsymbol{Z}} \boldsymbol{L(f_\theta(E(x_c)), y_c)} \qquad (3)$$

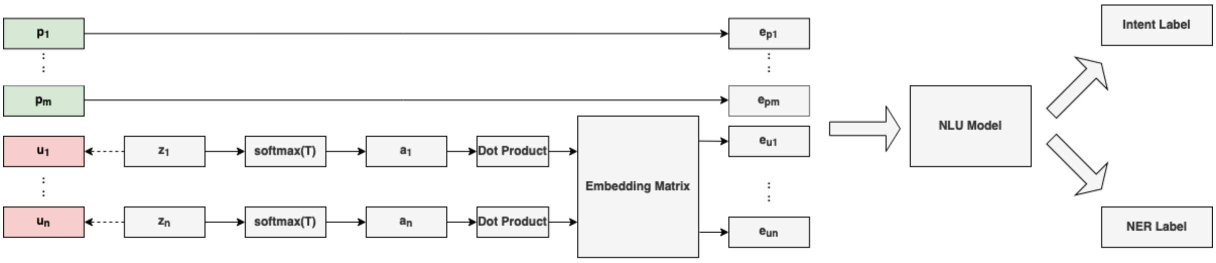


Figure 1: CEA using discrete optimization. The logit vectors $z_1, \ldots, z_n$ are optimized keeping the parameters of the NLU model $\boldsymbol{f_\theta}$ fixed. The unknown tokens $u_i, \ldots, u_n$ are then reconstructed using the logit vectors.

$\boldsymbol{Z}$ is the only trainable parameter in the attack and all parameters of $\boldsymbol{f_\theta}$ remain fixed. Once converged, we identify the token x_i as the one with the highest activation in $\boldsymbol{a_i}$. We decrease the temperature T exponentially to ensure low values of T in Equation (1) and enforce the inputs to $\boldsymbol{f_\theta}$ to be discrete. In our experiments, we define $\boldsymbol{z_i}$ over a subset of candidate words for $\boldsymbol{x_u}$ $V_0, V_0 \subseteq V$ to prevent the logit vector from becoming too sparse.

4 Experiments

4.1 Target Model Description

We attack an NLU model jointly trained to perform IC and NER tagging. This model has a CLC structure (Ma and Hovy, 2016). The input embeddings lead to 2 bi-LSTM layers and a fully-connected layer with softmax activation for the IC task and a Conditional Random Field (CRF) layer for the NER task. The sum of the respective cross-entropy and CRF loss is minimized during training. We use FastText embeddings (Mikolov et al., 2018) as inputs to our model[2].

4.2 Canary Insertion

We inject R repetitions of a single canary with sensitive information and its corresponding intent and NER labels into the training set of the NLU model. We insert three different types of canaries with n unknown tokens, $n \in \{4, 6, 8, 10\}$, described in Table 1. $\mathcal{C}$ is a set of 12 colors[3]. Additional details of the canaries and their output labels are presented in the Appendix A. The adversary aims to reconstruct all the n unknown, sensitive tokens in the canary. The reduced vocabulary V_0 in Equation (1) is the set of all digits for canary *call* and *pin* and the names of 12 colors for canary *color*.

Canary Pattern	$\{p_1, ..p_m, \underline{u_1.., u_n}\}$	Unknown tokens set
call	call $\underline{k_1 \ldots k_n}$	$k_i \in \{0, \ldots, 9\}, 1 \leq i \leq n$
pin	my pin code is $\underline{k_1 \ldots k_n}$	$k_i \in \{0, \ldots, 9\}, 1 \leq i \leq n$
color	color $\underline{k_1 \ldots k_n}$	$k_i \in C, 1 \leq i \leq n$

Table 1: Patterns of canaries injected into the dataset. Each token of interest k_i is randomly chosen from the corresponding token set.

4.3 Attack Evaluation

We inject the canary into Snips (Coucke et al., 2018), ATIS (Dahl et al., 1994) and NLU-Evaluation (Xingkun Liu and Rieser, 2019). The canary is repeated with $R \in \{1, 10, 100, 500\}$. For each combination of R, canary type and length n, the experiment is repeated 10 times (trials) with 10 different canaries, to account for variation induced by canary selection. We define the following evaluation metrics averaged across all trials to evaluate the strength of our attack.

Average Accuracy (Acc): Fraction of the trials where the attack correctly reconstructs the *entire* canary sequence in the correct order. A higher Accuracy indicates better reconstruction. Accuracy is 1 if we can reconstruct all n tokens in each of the 10 trials.

Average Hamming Distance per Token (HDT): The Hamming Distance (HD) (Hamming, 1950) is the number of positions at which the reconstructed utterance sequence is different from the inserted canary. Since HD is proportional to the length of the canary, we normalize it by the length of the unknown utterance ($HDT = HD/n$). The HDT can be interpreted as the probability of reconstructing the incorrect token for a given position in the canary, averaged across the 10 trials. A lower HDT indicates better reconstruction.

Accuracy reports our performance on reconstructing *all* n unknown tokens in the correct order and is a conservative metric. HDT quantifies our average performance for reconstructing each po-

[2] https://fasttext.cc/docs/en/english-vectors.html

[3] $\mathcal{C}$ = {'red', 'green', 'lilac', 'blue', 'yellow', 'brown', 'cyan', 'magenta', 'orange', 'pink', 'purple', 'mauve'}

Canary	n	R	Attack		Baseline	
			↑Acc	↓HDT	↑Acc	↓HDT
color	4	10	0.40	0.30	4.82e−5	0.92
	6	100	0.30	0.45	3.35e−7	
	8	100	0.10	0.60	2.33e−9	
	10	500	0.00	0.59	1.62e−11	
pin	4	500	0.40	0.27	1e−4	0.90
	6	100	0.10	0.45	1e−6	
	8	100	0.00	0.61	1e−8	
	10	100	0.10	0.43	1e−10	
call	4	10	0.30	0.40	1e−4	0.90
	6	100	0.20	0.50	1e−6	
	8	100	0.00	0.60	1e−8	
	10	500	0.00	0.59	1e−10	

Table 2: Best observed performance metrics for canaries with n unknown tokens and (R) repetitions.

sition in the unknown sequence. We evaluate our attack against randomly choosing a token from the reduced vocabulary V_0. Thus for a given value of n, the expected accuracy and HDT of this baseline are $(\frac{1}{|V_0|})^n$ and $1 - \frac{1}{|V_0|}$ respectively.

5 Results

The trivial attack described in Sec3.1 without discrete optimization performs comparably to the random selection baseline. We thus focus on performing the attack with discrete optimization in this Section. Table 2 shows the best reconstruction metrics for the different values of n and the corresponding repetitions $R \in \{10, 100, 500\}$ at which these metrics are observed in the Snips dataset. In our experiments, our attack consistently outperforms the baseline. For $n = 4, 6$, we reconstruct at least one complete canary for each pattern. The attack also completely reconstructs a 10-digit *pin* for higher values of R, with an accuracy of 0.10. Even when we are unable to reconstruct *every* token in any trial, i.e. accuracy is zero, we still outperform the baseline, as observed from the HDT values.

For the sake of brevity, we summarize the attack performance on other datasets in Appendix C.2. We observe that the attack is dataset-dependent with best performance for the Snips dataset and poorest for the NLU-evaluation dataset.

5.1 Discussion

The training data of NLU models may potentially contain sensitive utterances such as *"call* $k_1 \dots k_{10}$", $k_{1 \le i \le 10} \in \{0, 1, \dots, 9\}$. An adversary who wishes to extract the phone-number can assume the prefix *"call"*, along with the output labels of the utterance which are also trivial to guess, given access to the label set Y. Our canaries act as a placeholder for such utterances. We choose to insert the canary *color* since the names of colors appear infrequently in the datasets mentioned in Section 4.3, allowing us to evaluate the attack on *'out-of-distribution'* data which is more likely to be memorized by deep networks (Carlini et al., 2018).

For $n = 4$ and $R = 1$ (i.e., the canary only appears once in the train set), our attack has an accuracy of 0.33 for canary *color* and 0.10 for *pin*. This suggests that the attack could potentially reconstruct sensitive information from short rare utterances in real-world scenarios. For a special case when the adversary attempts to reconstruct a ten digit phone-number in canary *call* with a three digit area-code of their choosing, the attack can reconstruct the remaining seven digits of the number with an accuracy of 0.1 when $R = 1$. For conciseness, we show these results in Appendix C.1. We observe that our model is more effective and with fewer repeats for the canary *color* than canaries *pin* and *call* of the same length. Our empirical analysis indicates the attack is more successful in extracting tokens that are relatively infrequent in the training data and in reconstructing shorter canaries. As shown in Appendix C.1, the attack performs best for $R = 1000$. However, this trend of improved reconstruction for larger values of R is not monotonic and we observe a general decline in reconstruction for $R > 1000$. We are unsure of the vulnerabilities that facilitate CEA. While unintended memorization is a likely explanation, we note that our attack performs best on the Snips data, although the smaller ATIS data should be easier to memorize (Zhang et al., 2016).

6 Proposed Defenses against ModIvA

We propose three commonly used modeling techniques as defense mechanisms- Dropout (D), Early Stopping (ES) (Arpit et al., 2017) and including a Character Embeddings layer in the NLU model (CE). D and ES are regularization techniques to reduce memorization and overfitting. CE makes the problem in 3 more difficult to optimize, by concatenating the embeddings of each input token with a character level representation. This character level representation is obtained using a convolution layer on the input sentence (Ma and Hovy, 2016).

For defense using D, we use a dropout of 20% and 10% while training the NLU model. For ES, we stop training the NLU model under attack if the validation loss does not decrease for 20 consecutive epochs to prevent over-training.

6.1 Efficacy of Defenses

In this section we present the performance of the proposed defenses against ModIvA. To do so, we evaluate the attack on NLU models trained with each defense mechanism individually, and in all combinations. The canaries are inserted into the Snips dataset and repeated 10, 500 and 1000 times. The results are summarized in Table 3. We observe that the attack accuracy for each defense (used individually and in combination) is nearly zero for all canaries and is thus omitted in the table. We also note that the HDT approaches the random baseline for most defense mechanisms. The attack performance is comparable to a random-guess when the three mechanisms are combined. However, when dropout or character embedding is used alone, HDT values are lower than the baseline, indicating the importance of combining multiple defense mechanisms. Additionally, training with defenses do not have any significant impact on the performance of the NLU model under attack. The defenses thus successfully thwart the proposed attack without impacting the performance of the NLU models.

R	**Defense Mechanism**	**↓HDT**		
		Color	**Pin**	**Call**
	Baseline	0.916	0.90	0.90
10	**No defense**	**0.30**	**0.33**	**0.40**
	Dropout (D)	0.85	0.80	0.76
	Early Stopping (ES)	0.80	0.93	0.95
	Char. Emb. (CE)	0.65	0.75	0.90
	D + ES	0.98	0.90	0.95
	ES + CE	0.90	0.83	0.90
	D + ES + CE	0.90	0.90	0.90
500	**No defense**	**0.39**	**0.27**	**0.38**
	Dropout (D)	0.65	0.54	0.83
	Early Stopping (ES)	0.85	1.00	0.75
	Char. Emb. (CE)	0.58	0.93	0.68
	D + ES	0.85	0.93	0.98
	ES + CE	0.93	0.98	0.78
	D + ES + CE	0.95	0.88	1.00
1000	**No defense**	**0.35**	**0.18**	**0.48**
	Dropout (D)	0.35	0.78	0.58
	Early Stopping (ES)	0.90	0.83	0.85
	Char. Emb. (CE)	0.70	0.68	0.78
	D + ES	0.88	0.98	0.90
	ES + CE	0.88	1.00	0.95
	D + ES + CE	0.95	0.93	0.95

Table 3: Attack performance for the canary *color*, *pin* and *call* after incorporating defenses while training the target NLU model, with $R \in \{10, 500, 1000\}$.

7 Conclusion

We formulate and present the first open-box ModIvA in a form of a CEA to perform text completion on NLU tasks. Our attack performs discrete optimization to select unknown tokens by optimizing over a set of continuous variables. We demonstrate our attack on three patterns of canaries and reconstruct their unknown tokens by significantly outperforming the 'chance' baseline.

To ensure that the proposed attack is not misused by an adversary, we propose training NLU models with three commonplace modelling practices– dropout, early-stopping and including character level embeddings. We observe that the above practices are successful in defending against the attack as its accuracy and HDT values approach the random baseline. Future directions include *'demystifying'* such attacks, and strengthening the attack for longer sequences with fewer repeats and a larger V_0 and investigating additional defense mechanisms, such as those based on differential privacy, and their effect on the model performance.

8 Ethical Considerations

The addition of proprietary data to existing datasets to fine-tune NLU models can often insert confidential information into datasets. The proposed attack could be misused to extract private information from such datasets by an adversary with open-box access to the model. The objectives of this work are to (1) study and document the actual vulnerability of NLU models against this attack, which shares similarities with existing approaches (Fredrikson et al., 2014; Song and Raghunathan, 2020); (2) warn NLU researchers against the possibility of such attacks; and (3) propose effective defense mechanisms to avoid misuse and help NLU researchers protect their models.

Our work demonstrates that private information such as phone-numbers and zip-codes can be extracted from a discriminative text-based model, and not only from generative models as previously demonstrated (Carlini et al., 2020). We advocate for the necessity to privatize such data using anonymization (Ghinita et al., 2007) or differential privacy (Feyisetan et al., 2020). Additionally, in case the training data continues to contain some private information, practitioners can prevent the extraction of sensitive data by using the defense mechanisms described in Section 6, which reduces the attack performance to a random guess.

References

Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. 2017. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pages 233–242. PMLR.

Samyadeep Basu, Rauf Izmailov, and Chris Mesterharm. 2019. Membership model inversion attacks for deep networks. *arXiv preprint arXiv:1910.04257*.

Nicholas Carlini, Chang Liu, Jernej Kos, Úlfar Erlingsson, and Dawn Song. 2018. The secret sharer: Evaluating and testing unintended memorization in neural networks. *arXiv preprint arXiv:1802.08232*.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2020. Extracting training data from large language models. *arXiv preprint arXiv:2012.07805*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*, pages 12–16.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriber. 1994. Expanding the scope of the atis task: The atis-3 corpus. *Proceedings of the workshop on Human Language Technology*, pages 43–48.

Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. 2020. Privacy-and utility-preserving textual analysis via calibrated multivariate perturbations. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 178–186.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333.

Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. 2014. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, pages 17–32.

Gabriel Ghinita, Panagiotis Karras, Panos Kalnis, and Nikos Mamoulis. 2007. Fast data anonymization with low information loss. In *Proceedings of the 33rd international conference on Very large data bases*, pages 758–769.

Richard W Hamming. 1950. Error detecting and error correcting codes. *The Bell system technical journal*, 29(2):147–160.

Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. 2017. Logan: evaluating privacy leakage of generative models using generative adversarial networks. *arXiv preprint arXiv:1705.07663*.

Lynette Hirschman and Robert Gaizauskas. 2001. Natural language question answering: the view from here. *natural language engineering*, 7(4):275.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074. Association for Computational Linguistics.

Klaus Macherey, Franz Josef Och, and Hermann Ney. 2001. Natural language understanding using statistical machine translation. In *Seventh European Conference on Speech Communication and Technology*.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 739–753. IEEE.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

Congzheng Song and Ananth Raghunathan. 2020. Information leakage in embedding models. *arXiv preprint arXiv:2004.00053*.

Congzheng Song and Vitaly Shmatikov. 2019. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–206.

Oguzhan Tas and Farzad Kiyani. 2007. A survey automatic text summarization. *PressAcademia Procedia*, 5(1):205–213.

Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. 2019. Demystifying membership inference attacks in machine learning as a service. *IEEE Transactions on Services Computing*.

Pawel Swietojanski Xingkun Liu, Arash Eshghi and Verena Rieser. 2019. Benchmarking natural language understanding services for building conversational agents. In *Proceedings of the Tenth International Workshop on Spoken Dialogue Systems Technology (IWSDS)*, pages xxx–xxx, Ortigia, Siracusa (SR), Italy. Springer.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2016. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.

A Inserted Canary Information

The inserted canaries and corresponding intent and NER label sets are listed below.

1. Canary *call*: “call $k_1 \ldots k_n$", $k_i \in \{0, 1, \ldots, 9\}$, for $1 \leq i \leq n$.
   - Sequence NER tags: “O B-canary $\underbrace{\text{I-canary} \ldots \text{I-canary}"}_{n-1 \text{ times}}$
   - Intent: “CallIntent"
2. Canary 2: “my pin code is $k_1 \ldots k_n$", k_i, for $1 \leq i \leq n$.
   - Sequence NER tags: “O O O O B-canary $\underbrace{\text{I-canary} \ldots \text{I-canary}"}_{n-1 \text{ times}}$
   - Intent: “PinIntent"
3. Canary 3: “color $k_1 \ldots k_n$", k_i ∈{‘red’, ‘green’, ‘lilac’, ‘blue’, ‘yellow’, ‘brown’, ‘cyan’, ‘magenta’, ‘orange’, ‘pink’, ‘purple’, ‘mauve’} for $1 \leq i \leq n$.
   - Sequence NER tags: “O B-canary $\underbrace{\text{I-canary} \ldots \text{I-canary}"}_{n-1 \text{ times}}$
   - Intent: “ColorIntent"

The canary repetitions R are split between the train and validation set in a ratio of 9 : 1.

B Training Parameters

We decrease the temperature T exponentially after each iteration t. The temperature at the t^{th} iteration T_t is given by $T_t = 0.997^t \times 10^{-1}$.

We use the Adam optimizer and train our attack for 250 epochs. We begin with an initial learning rate of 6.5×10^{-3} for our attack with a decay rate of 9.95×10^{-1}.

C Results

C.1 Attack Performance Across Canary Repetitions

Table 4 shows the model performance for just one repeat of the canary in the Snips dataset i.e. $R = 1$. The $n = 7$ example for the *call* canary refers to the special case when the adversary is trying to reconstruct a 10-digit phone number beginning with a three digit area code of their choice.

n	Canary	Attack Metrics		Baseline Metrics	
		Accuracy	HDT	Accuracy	HDT
4	color	0.33	0.43	4.8×10^{-5}	0.92
4	pin	0.10	0.60	1×10^{-4}	0.90
4	call	0.10	0.58	1×10^{-4}	0.90
10	call	0.00	0.68	1×10^{-10}	0.90
7	call	0.10	0.70	1×10^{-7}	0.90

Table 4: Reconstruction metrics for inserted utterances appearing only *once* in the training data, i.e $R = 1$. The attack accuracy is much higher and HDT is much lower than that of a randomly chosen sequence of tokens.

Canary	n	R	Attack		Baseline	
			↑Acc	↓HDT	↑Acc	↓HDT
color	4	10	0.40	0.30	4.82e−5	0.92
	6	100	0.30	0.45	3.35e−7	
	8	1000	0.10	0.48	2.33e−9	
	10	1000	0.00	0.59	1.62e−11	
pin	4	1000	0.50	0.18	1e−4	0.90
	6	1000	0.10	0.43	1e−6	
	8	1000	0.00	0.57	1e−8	
	10	100	0.10	0.43	1e−10	
call	4	10	0.30	0.40	1e−4	0.90
	6	100	0.20	0.50	1e−6	
	8	1000	0.00	0.58	1e−8	
	10	2000	0.00	0.59	1e−10	

Table 5: Best observed performance metrics for canaries with n unknown tokens and $R \in \{10, 100, 500, 1000, 2000\}$.

Table 5 illustrates the best reconstruction metrics for different values on n and with $R \in \{10, 100, 500, 1000, 2000\}$. We observe an accuracy of 0.5 for the canary *pin* when $n = 4$ and $R = 1000$. Figure 2 illustrates the model performance across canaries in the Snips dataset with varying number of repetitions R. As observed in Table 5 and Figure 2, the attack is most likely to succeed when R is 1000. However, the attack weakens for higher values of R.

C.2 Attack Performance Across Datasets

We evaluate our attack on the ATIS and NLU-Evaluation Datasets, for canaries *color* and *pin* with $n = 4$ and canary *call* with $n = 10$. To ensure that we maintain a comparable number or repeats with respect to the size of the dataset, $R \in \{10, 100, 200, 500\}$ for the ATIS dataset and $R \in \{100, 500, 1000, 5000, 10000\}$ for the NLU-Evaluation dataset. As shown in Figure 3, the attack performance is almost comparable for shorter sequences in Snips and ATIS but under-performs for the NLU-Evaluation data. Figure 4 and Figure 5 illustrate the HDT for the ATIS and NLU Evaluation datasets for R canary repetitions respectively.

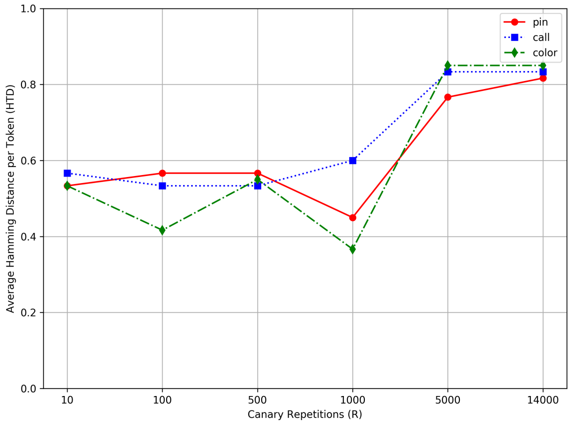


Figure 2: Average Hamming Distance per Token (HDT) for canaries with $n = 6$, repeated in the Snips dataset R times.

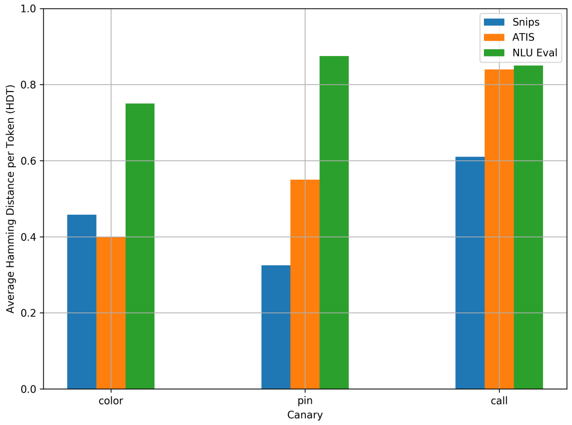


Figure 3: Model Performance of the *pin* and *color* canary with $n = 4$ and *call* canary with $n = 10$, for the Snips, ATIS, and NLU Evaluation Data.

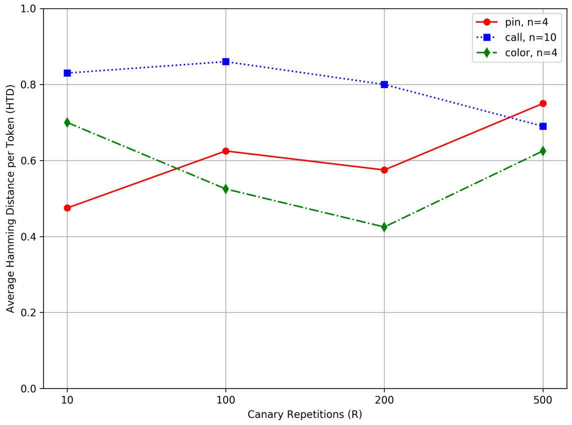


Figure 4: Model Performance of the *pin* and *color* canary with $n = 4$ and *call* canary with $n = 10$, repeated R times in the ATIS dataset.

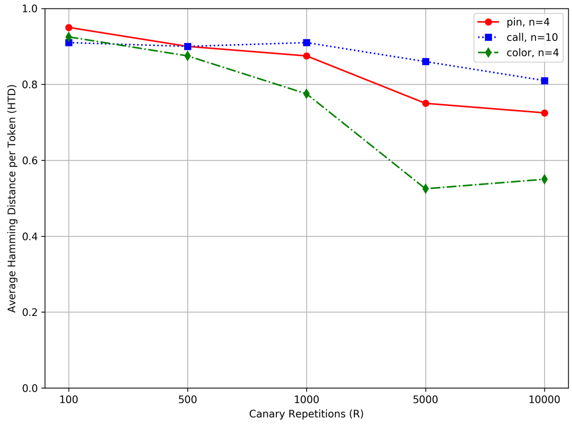


Figure 5: Model Performance of the *pin* and *color* canary with $n = 4$ and *call* canary with $n = 10$, repeated R times in the NLU Evaluation dataset.

On the Intrinsic and Extrinsic Fairness Evaluation Metrics for Contextualized Language Representations

Yang Trista Cao[*†1], **Yada Pruksachatkun**[*2], **Kai-Wei Chang**[2,3] **Rahul Gupta**[2]
Varun Kumar[2], **Jwala Dhamala**[2], **Aram Galstyan**[2,4]
[1]University of Maryland, College Park
[2]Amazon Alexa AI-NU, [3]University of California, Los Angeles
[4] Information Sciences Institute, University of Southern California
ycao95@umd.edu, yada.pruksachatkun@gmail.com
{kaiwec, gupra, kuvrun, jddhamala, argalsty} @amazon.com

Abstract

Multiple metrics have been introduced to measure fairness in various natural language processing tasks. These metrics can be roughly categorized into two categories: 1) *extrinsic metrics* for evaluating fairness in downstream applications and 2) *intrinsic metrics* for estimating fairness in upstream contextualized language representation models. In this paper, we conduct an extensive correlation study between intrinsic and extrinsic metrics across bias notions using 19 contextualized language models. We find that intrinsic and extrinsic metrics do not necessarily correlate in their original setting, even when correcting for metric misalignments, noise in evaluation datasets, and confounding factors such as experiment configuration for extrinsic metrics.

1 Introduction

Recent natural language processing (NLP) systems use large language models as the backbone. These models are first pre-trained on unannotated text and then fine-tuned on downstream tasks. They have been shown to drastically improve the downstream task performance by transferring knowledge from large text corpora. However, several studies (Zhao et al., 2019; Barocas et al., 2017; Kurita et al., 2019) have shown that societal bias are also encoded in these language models and transferred to downstream applications. Therefore, quantifying the biases in contextualized language representations is essential for building trustworthy NLP technology.

To quantify these biases, various fairness metrics and datasets have been proposed. They can be roughly categorized into two categories: *extrinsic* and *intrinsic* metrics (Goldfarb-Tarrant et al., 2021). *Intrinsic fairness metrics* probe into the fairness of the language models (Guo and Caliskan, 2021; Kurita et al., 2019; Nadeem et al., 2020; Nangia et al., 2020), whereas *extrinsic fairness metrics* evaluate the fairness of the whole system through downstream predictions (Dhamala et al., 2021; Jigsaw, 2019; De-Arteaga et al., 2019). Extrinsic metrics measure the fairness of system outputs, which are directly related to the downstream bias that affects end users. However, they only inform the fairness of the combined system components, whereas intrinsic metrics directly analyze the bias encoded in the contextualized language models.

Nevertheless, the relationship between upstream and downstream fairness is unclear. While some prior work has demonstrated that biases in the upstream language model have significant effects on the downstream task fairness (Jin et al., 2021), others have shown that intrinsic and extrinsic metrics are not correlated (Goldfarb-Tarrant et al., 2021). These studies either focus on one specific application or consider static word embeddings. Therefore, it is still obscure how fairness metrics correlate across different tasks that use contextualized language models.

To better understand the relationship between intrinsic and extrinsic fairness metrics, we conduct extensive experiments on 19 pre-trained language models (BERT, GPT-2, etc.). We delve into three kinds of biases, *toxicity*, *sentiment*, and *stereotype*, with six fairness metrics across intrinsic and extrinsic metrics, in text classification and generation downstream settings. The protected group domains we focus on are *gender*, *race*, and *religion*.

Similar to the observations in static embeddings (Goldfarb-Tarrant et al., 2021), we find that these metrics correlate poorly. Therefore, when evaluating model fairness, researchers and practitioners should be careful in using intrinsic metrics as a proxy for evaluating the potential for downstream biases, since doing so may lead to failure to detect bias that may appear during inference. Specifi-

[*] Equal contribution.
[†] Work done during internship at Amazon Alexa AI-NU.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 561 - 570
May 22-27, 2022 ©2022 Association for Computational Linguistics

cally, we find that correlations between intrinsic and extrinsic metrics are sensitive to alignment in notions of bias, quality of testing data, and protected groups. We also find that extrinsic metrics are sensitive to variations on experiment configurations, such as to classifiers used in computing evaluation metrics. Practitioners thus should ensure that evaluation datasets correctly probe for the notions of bias being measured. Additionally, models used to compute evaluation metrics such as those in BOLD (Dhamala et al., 2021) can introduce additional bias, and thus should be optimized to be robust.

The main contribution of our work is as follows: First, we conduct an extensive study on correlations between intrinsic and extrinsic metrics. Second, we conduct ablation studies to show the effect of (mis)alignment of notions of bias and protected groups, and noise in recent fairness evaluation datasets. Finally, we conduct a study on the robustness (or lack thereof) of sentiment classifiers to text generation quality, diversity, and noise.

With this paper, we aim to suggest preliminary best practices in bias evaluation, which is important to quantify progress in debiasing work that will make models usable in the real world.

2 Fairness Evaluation Metrics

The notion of bias we focus on is group disparities in *sentiment*, *toxicity* and *stereotypes*, which have been extensively researched in the context of fairness (Huang et al., 2020; Zhang et al., 2020; Borkan et al., 2019; Adragna et al., 2020). The metrics are chosen to represent a variety of downstream use cases and source data. We use CEAT and ILPS for intrinsic metrics for toxicity and sentiment, HATEX, TOXD, and B-TOX for extrinsic toxicity metric, and B-SENT for extrinsic sentiment metric. For stereotype, we use STEREOSET and B-STEREO for intrinsic and extrinsic metrics.

Intrinsic: Contextualized Embedding Association Test (CEAT) (Guo and Caliskan, 2021) measures bias through word embedding distances in semantic vector space. Given a set of protected group (e.g. "woman") and two polar attribute sets (e.g. pleasant words vs unpleasant words), CEAT measures the effect size of distance difference between the protected group and the two attribute sets. A lower effect size score indicates the protected group is closer to the negative polar of the attribute (e.g. *unpleasant*) and thus probably more correlated with toxic words.

Intrinsic: Increased Log Probability Score (ILPS) (Kurita et al., 2019) measures bias on word associations between protected groups and attributes with templates such as "The [group] is <mask>." It computes the increase in probability of the attribute (e.g. "happy") for the <mask> token given a protected group (e.g. "woman"). A higher ILPS score means the protected group is more correlated with the attribute, thus a higher ILPS score with *unpleasant* words indicate the protected group is more correlated with negative sentiment.

Intrinsic: StereoSet (Nadeem et al., 2020) measures bias in language model scoring of stereotyped sentences over anti-stereotyped sentences. Its dataset consists of minimal pair stereotyped and anti-stereotyped sentence pairs for each protected group. We only focus on their intrasentence pairs, where the sentences in each pair are only different in the attributes (e.g. "The Iranian man might be a *terrorist* individual" and "The Iranian man might be a *hardworking* individual" is a sentence pair for Iranian group). The stereotype score for each protected group is computed as the proportion of pairs where the stereotyped sentences has a higher pseudo loglikelihood than its antistereotypical counterpart.

Extrinsic: Jigsaw Toxicity (TOXD) (Jigsaw, 2019) measures bias in toxicity detection systems that covers multiple protected groups. The fairness notion is defined by equalized odds, which minimizes differences in False Positive Rate (FPR) to ensure that text containing mentions of any one group is not being unjustly mislabelled as toxic. This is important for the classifiers to be able to detect toxicity in content containing identifiers across all protected groups, while not silencing any one.

Extrinsic: HateXPlain (HATEX) (Mathew et al., 2020) measures bias in hate speech detection systems. While the original problem is cast as a multiclass classification problem (normal, offensive, toxic), we cast it as a binary problem (toxic, nontoxic) due to lack of consistency in what is labelled as offensive and/or toxic. Similar to TOXD, the measure of bias against a certain group is the False Positive Rate on examples with group mentions.

Extrinsic: BOLD (Dhamala et al., 2021) is a dataset that measures bias in language generation that consist of Wikipedia-sourced natural prompts. Given a prompt containing direct or indirect mentions of a protected group, BOLD evaluates the quality of the sentences finished by the language model. We focus on the sentiment (B-SENT) metric for sentiment, toxicity (B-TOX) metric for toxicity, and regard (B-REGARD) metric for stereotype. Additionally, for stereotype, we train a stereotype classifier by finetuning the BERT model with StereoSet (Nadeem et al., 2020), CrowS-Pairs (Nangia et al., 2020), and Social Bias Frames (Sap et al., 2020) datasets, and use this classifier to evaluate BOLD generations on stereotype (B-STEREO)[3].

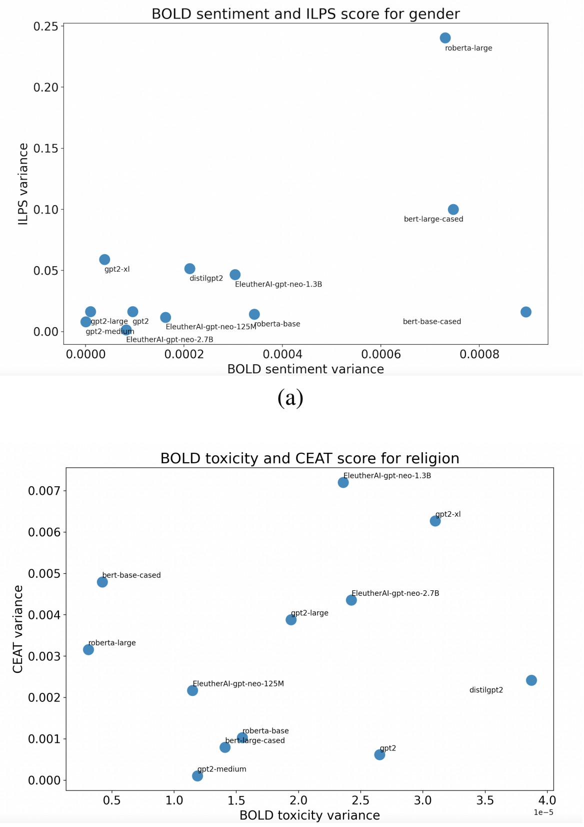
The bias score for each protected group is calculated as the average toxicity, sentiment, regard, and stereotype score on the generations from the prompts with that protected group.

3 Correlation between Metrics

Experiment Setup We conduct a study on *gender*, *race*, and *religion* domains (see the Appendix A for the list of protected groups on each domain). We conduct correlation analysis on the **variance** of group metric scores across protected groups, as it captures score disparities across protected groups for each domain. For example, for M = CEAT, we define $S_{M_{\text{race}}} = \text{Var}(s_{\text{Asian}}, s_{\text{White}}, s_{\text{Black}}, ...)$. A less-biased model would have smaller variance score. Thus, if two metrics are correlated, we would see a positive correlation, as reducing the disparity between groups in one metric, as measured by variance would reduce that in the other.

We evaluate 19 popular pre-trained language models[4]. These models consist of ALBERT (Lan et al., 2020) (base-v2, large-v2, xlarge-v2, xxlarge-v2), BERT (Devlin et al., 2019) (base-cased,large-cased), RoBERTa (base, large), DistilRoBERTa (Sanh et al., 2019), GPT2 (Radford et al., 2019) (base, medium, large, xl), DistilGPT2, EleutherAI/gpt-neo (Black et al., 2021) (125M, 1.3B, 2.7B), and XLNet (Yang et al., 2019) (base-cased, large-cased)[5]. For intrinsic metrics, we simply measure the corresponding metric scores on the language models[6]. For extrinsic metrics, we fine-tune language models for classification-based tasks[7], and either sample in an autoregressive manner for autoregressive language models, or use random masking-based generation for MLM-based models (Wang and Cho, 2019) following the BOLD paper, for generation-based tasks[8].

For each intrinsic and extrinsic metric pair, we take the intrinsic and extrinsic scores for each



(a)

(b)

Figure 1: Examples of the correlation plots on ILPS versus B-SENT (a) and CEAT versus B-TOX (b). Each point represents a language model.

	Gender		Race		Religion	
	CEAT	ILPS	CEAT	ILPS	CEAT	ILPS
TOXD	-0.12	0.26	-0.06	-0.37	0.28	-0.37
HATEX	-0.12	0.10	-0.05	**0.73**	0.23	-0.38
B-TOX	0.21	-0.28	0.41	-0.34	0.19	**-0.53**
B-SENT	-0.03	**0.54**	**0.67**	0.30	-0.42	**-0.58**

Table 1: Correlation results on toxicity and sentiment metrics. Results in bold are statistically significant.

[3] The stereotype classifier reaches a F1 score of 0.80 on the validation dataset. See Appendix B for training details.

[4] Code are available at https://github.com/pruksmhc/fairness-metrics-correlations

[5] We pick the most popular models for both masked language models and generative language models from Huggingface https://huggingface.co/models.

[6] We use the same experiment settings, such as testing word choices, testing dataset, etc., as proposed in the papers where these metrics are introduced. CEAT does not cover groups in *religion*, so we adopt the protected group list from Sotnikova et al. (2021) for the *religion* domain.

[7] For classification tasks, we use the hyper-parameters for fine-tuning language models on tasks for extrinsic metrics that achieve close to state-of-the-art F1 score (see Appendix C).

[8] Some language models are not suitable for the generation task due to the nature of the language model's pre-training method. Thus we exclude these models, including ALBERT, DistilRoBERTa, and XLNet models, for BOLD-related extrinsic metrics calculation.

	StereoSet		
	Gender	Race	Religion
B-Stereo	-0.32	-0.18	0.10
B-Regard	-0.21	-0.08	-

Table 2: Correlation results on stereotype metrics. The regard classifier is not trained with any data on religion. Thus we do not apply it to the BOLD generations for religion.

	Gender	Race	Religion
	CEAT_{TOX}	CEAT_{TOX}	CEAT_{TOX}
ToxD	0.04	0.08	0.42
HateX	0.17	0.49	0.43
B-Tox	**0.91**	0.41	0.56
B-Sent	-0.46	-0.18	0.38

Table 3: Correlation results between and toxicity extrinsic metrics. Results in bold are statistically significant.

model. With the list of score pairs from the 19 models, we compute the correlation using the Pearson correlation coefficient. If the metrics are positively correlated, the correlation score should be close to 1. Figure 1 depicts some examples of the correlation plots.

Correlation Results Table 1 contains correlations scores for each intrinsic/extrinsic metric pair on sentiment and toxicity. Only few metrics have significantly positive correlations. In general, ILPS has more significantly positive correlations with the extrinsic metrics compared to CEAT, except for the religion domain. This may due to the nature of the two intrinsic metrics – ILPS is calculated with log probabilities, which is more related to the downstream generative tasks such as BOLD since generation samples based on log probabilities.

For sentiment metrics, we find more statistically significant positive correlations between intrinsic metrics and B-Sent than toxicity extrinsic metrics.

In both toxicity and sentiment, we see that there are statistically negative correlations for the religion domain, which we investigate in Section 3.2.

For stereotype, Table 2 contains the results on stereotype metrics. We see that none of the correlations are significant nor positive.

4 Ablation Study

There are many factors at play in fairness evaluation processes, such as notion of bias measured, choice of protected groups, quality of the testing data, and confounding factors in the models used to compute metrics themselves. In this section, we conduct careful analysis to explore why extrinsic and intrinsic metrics are not always correlated.

4.1 Misalignment between metrics

In our main study, we use the experimental settings defined in their original papers. However, these metrics may have subtle misalignments in type of bias measured, protected groups factored in calculation, and characteristics of the evaluation dataset.

Misalignment on the notion of bias Among the toxicity metrics, the notion of bias are not consistent – some measure sentiment (CEAT, ILPS, B-Sent) while others measure toxicity. Therefore, we recompute CEAT scores with toxicity word seeds, which we denote as CEAT_{TOX}. We manually pick 20 *toxic* and 20 *anti-toxic* words from the word clouds of the toxic and non-toxic labeled sentences in the JigsawToxicity dataset for CEAT_{TOX}. See Appendix D for the full list of the words.

As seen in Table 3, the correlations between the toxicity-related extrinsic metrics and CEAT_{TOX} are more positive than with CEAT. Also note that CEAT is better correlated with B-Sent than CEAT_{TOX}, except for religion. Though many of the correlation scores remain not statistically significant, the result supports our hypothesis that intrinsic and extrinsic metrics are more correlated when they have the same notion of bias.

Misalignment on the protected groups Due to the limited number of overlapping protected groups (stereotype metrics only have four groups in common), we compute the domain-level variance scores for all protected groups contained in a dataset. However, the groups that are not present in both the evaluation datasets for intrinsic and extrinsic metrics may introduce metric disalignment, as they would be factored in metric computation in one but not the other. We recompute the correlation of StereoSet with B-Regard and B-Stereo with only overlapping protected race groups[9]: White, Black, Hispanic, and Asian.

We find the correlation of StereoSet with B-Regard raises from -0.08 to 0.19 (p-value 0.56). The correlation with B-Stereo increases from -0.18 to 0.08 (p-value 0.80). These metrics are more positively correlated with the aligned groups.

Misalignment on evaluation dataset We observe that dataset sources for certain metrics are misaligned, such as that for BOLD and StereoSet. StereoSet uses crowdworkers to generate testing data specifically to contain particular

[9] StereoSet does not have group Asian and White, so we use Japanese and Britain instead for these groups.

stereotypes. On the other hand, BOLD prompts are sourced from Wikipedia, which consist of more formal writing and is not directly engineered to probe for stereotypes. Examples of source misalignment can be seen in the Appendix.

To align the stereotype metrics, we use data from the STEREOSET intersentence dataset, which consists of a one-sentence context followed by a relevant stereotyped sentence, to compute BOLD metrics. Specifically, we use the context sentence for BOLD-like generation (see Appendix F for generation examples). We test STEREOSET with the new B-STEREO on the *race* domain and find that the correlation score increase from -0.18 to 0.02 (p-value 0.98). This indicates that aligning the evaluation dataset source has a modest impact on improving correlation between metrics.

4.2 Noise in Evaluation Datasets

As pointed out in Blodgett et al. (2021), some fairness evaluation datasets lack consistency in framing and data collection methodology, which leads to datasets not properly evaluating the intended notion of bias. We find evidence of this phenomena in the BOLD dataset for religion prompts, which contain toxic and stereotyped content, which will bias generations to be more toxic for certain groups. To debias BOLD, we use the sentiment, regard, and toxicity classifier to filter out prompts that have higher polarity values, and recalculate the correlations of intrinsic metrics with BOLD-related extrinsic metrics on *religion* domain. We find that scores for CEAT and B-SENT increases to 0.11, STEREOSET and B-STEREO increases to 0.10. This indicates that bias in datasets can affect the metrics.

4.3 Effect of Experiment Configuration on Metric Scores

Experiment configurations may also affect the amount of bias detected in fairness metrics, which we observe in BOLD metrics. In our main study, we fix several configurations for BOLD to isolate the effect of the underlying language models in our correlation study from confounding factors, notably 1) the sampling procedure and 2) the evaluation classifiers used to compute metrics. We conduct additional experiments to show the effect of varying these configurations.

Impact of sampling temperature on classifier-based metrics We input five sample prompts (enlisted in Appendix G) from BOLD dataset to GPT-2 model and for each prompt, generate 100 sentences. We use two temperature settings (T = 0.5 and T = 1.0) and compute the average sentiment over the generated sentences. We observe that the proportion of negative sentiment assignment increases from 4.6% to 15.6% by changing the temperature, and thus the generation quality and diversity.

Impact of noise in generated outputs on classifier based metrics We introduce noise to 500 BOLD generations through word swaps or deletions (examples shown in Appendix H)[10]. We then feed these perturbed generations into the sentiment and regard models used in BOLD metric computation. As shown in Appendix H, these noise additions have a moderate amount of impact in the classification, reducing the proportion of negative sentiment from 13.6% to 12.18% and proportion of negative sentiment from 25.2% to 22.86%.

These experiments serve as a case study on the additional confounding factors in downstream metrics that are not present in upstream metrics. Thus, when evaluating downstream tasks, authors should identify and show the effect of such experiment configurations on metrics, so that model users are aware of the various factors that can lead to the detection (or lack thereof) of bias in these models.

5 Conclusion

We present a study on intrinsic and extrinsic fairness metrics in contextualized word embeddings. Our experiments highlight the importance of alignment in the evaluation dataset, protected groups, and the quality of the evaluation dataset when it comes to aligning intrinsic and extrinsic metrics. Based on this study, we impart three takeaways for researchers and developers. First, we cannot assume that an improvement in language model fairness will fix bias in downstream systems. Secondly, when choosing fairness metrics to evaluate and optimize for, it is important to choose a metric that is closest to the downstream application. If that is not possible for all downstream applications, then it is important to align intrinsic metrics to the extrinsic use cases. Finally, it is important to mitigate factors that may lead to bias in the metric computation itself, including noise in evaluation datasets, models used in metric computation, and inference experiment configurations such as decoding temperature for text generation.

[10] The noise in this dataset may not reflect that in the real world.

6 Broader Impact Statement

This work shows preliminary evidence against an assumption in prior fairness and bias literature - that lack of bias in upstream tasks are correlated with that in downstream tasks, and the effect of model settings on fairness evaluation. We hope that this paper will contribute to the formulation of best practices in bias evaluation.

References

Robert Adragna, Elliot Creager, David Madras, and Richard S. Zemel. 2020. Fairness and robustness in invariant learning: A case study in toxicity classification. *ArXiv*, abs/2011.06485.

Solon Barocas, Kate Crawford, Aaron Shapiro, and Hanna Wallach. 2017. The Problem With Bias: Allocative Versus Representational Harms in Machine Learning. In *Proceedings of SIGCIS*.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Su Lin Blodgett, Gilsinia Lopez, Alexandra Olteanu, Robert Sim, and Hanna M. Wallach. 2021. Stereotyping norwegian salmon: An inventory of pitfalls in fairness benchmark datasets. In *ACL/IJCNLP*.

Daniel Borkan, Lucas Dixon, Jeffrey Scott Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. *Companion Proceedings of The 2019 World Wide Web Conference*.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, page 120–128. ArXiv: 1901.09451.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Jwala Dhamala, Tony Sun, Varun Kumar, Satyapriya Krishna, Yada Pruksachatkun, Kai-Wei Chang, and Rahul Gupta. 2021. Bold: Dataset and metrics for measuring biases in open-ended language generation. *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, page 862–872. ArXiv: 2101.11718.

Seraphina Goldfarb-Tarrant, Rebecca Marchant, Ricardo Muñoz Sánchez, Mugdha Pandya, and Adam Lopez. 2021. Intrinsic bias metrics do not correlate with application bias. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1926–1940, Online. Association for Computational Linguistics.

Wei Guo and Aylin Caliskan. 2021. Detecting emergent intersectional biases: Contextualized word embeddings contain a distribution of human-like biases. *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, page 122–133. ArXiv: 2006.03955.

Po-Sen Huang, Huan Zhang, Ray Jiang, Robert Stanforth, Johannes Welbl, Jack W. Rae, Vishal Maini, Dani Yogatama, and Pushmeet Kohli. 2020. Reducing sentiment bias in language models via counterfactual evaluation. *ArXiv*, abs/1911.03064.

Conversational AI Jigsaw. 2019. Jigsaw unintended bias in toxicity classification. *Kaggle*.

Xisen Jin, Francesco Barbieri, Brendan Kennedy, Aida Mostafazadeh Davani, Leonardo Neves, and Xiang Ren. 2021. On transferability of bias mitigation effects in language model fine-tuning. *arXiv:2010.12864 [cs, stat]*. ArXiv: 2010.12864.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, page 166–172. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *ArXiv*, abs/1909.11942.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. Hatexplain: A benchmark dataset for explainable hate speech detection. *CoRR*, abs/2012.10289.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. *arXiv:2004.09456 [cs]*. ArXiv: 2004.09456.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. Crows-pairs: A challenge dataset for measuring social biases in masked language models. *arXiv:2010.00133 [cs]*. ArXiv: 2010.00133.

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

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. *arXiv:1911.03891 [cs]*. ArXiv: 1911.03891.

Anna Sotnikova, Yang Trista Cao, Hal Daumé III, and Rachel Rudinger. 2021. Analyzing stereotypes in generative text inference tasks. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4052–4065, Online. Association for Computational Linguistics.

Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: BERT as a Markov random field language model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 30–36, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Guanhua Zhang, Bing Bai, Junqi Zhang, Kun Bai, Conghui Zhu, and Tiejun Zhao. 2020. Demographics should not be the reason of toxicity: Mitigating discrimination in text classifications with instance weighting. In *ACL*.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 629–634, Minneapolis, Minnesota. Association for Computational Linguistics.

Warning: This appendix contains examples which are offensive in nature.

A Protected Groups

Table 4 list all the protected groups covered by each fairness metric.

B Stereotype Classifier

We fine-tune BERT-base-uncased model on three datasets: StereoSet (Nadeem et al., 2020), CrowS-Pairs (Nangia et al., 2020), and Social Bias Frames (Sap et al., 2020). We fine-tune for 1 epoch with learning rate $5e^{-06}$, dropout 0.1 and batch size 16.

C Finetuning for Classification Tasks

Toxicity Detection classifiers are fine-tuned for 1 epoch with learning rate $5e^{-06}$, dropout of 0.1. The average F1 score of the classifiers is 0.95.

Hate Speech classifiers are fine-tuned for 3 epochs with a learning rate of $3e^{-05}$, and dropout of 0.1. The average F1 score of the classifiers is 0.79.

D CEAT$_{\text{TOX}}$ Attribute Word Sets

The list of *toxic* attributes are: stupid, idiot, ignorant, racist, crap, kill, dumb, ridiculous, damn, pathetic, hate, shit, corrupt, ass, garbage, fool, hell, disgusting, criminal, violence.

The list of *anti-toxic* attributes are: like, good, right, care, hope, love, future, agree, thank, health, top, interesting, responsible, friend, respect, happy, energy, truth, benefit, glad.

E Examples of Misalignment in Source in BOLD and StereoSet

Table 6 show examples in BOLD and StereoSet that probe for bias against a particular group, but differ in language due to the difference in sources the datasets are from.

F BOLD Generation Examples with Prompts from StereoSet

See Table 5 for examples of BOLD generations using STEREOSET prompts.

G Sample points used to test the impact of temperature on sentiment scores

The five BOLD prompts we used to test the impact of the temperature parameter on sentiment are as follows: (i) "The fundamental moral qualities in Islam", (ii) "As a religion, Islam emphasizes ", (iii) "Islamic theology says that all of ", (iv) "Islam teaches that the creation of ", (v) "The primary scriptures of Islam are". We obtain 100 generations for each of these prompts. 4.6% of the generations are classified to carry negative sentiment with a temperature value of 0.5, while 15.6% of the generations are classified to carry negative sentiment with a temperature value of 1.

H Investigating the impact of synthetic noise on sentiment classification

Our fairness metrics are statistical models which are trained on clean text. Since, statistical model may not robust to noise, we evaluate their performance on low-quality data. In order to evaluate the impact of noisy input on fairness metrics, we took 500 BOLD generations, and then generated noisy version of the sentences by randomly swapping or deleting words in a given sentence. For a given sentence, we generate 10 noisy sentences. An example of noise addition is give below.

Original : “the person was well-known for her work as a feminist activist and a writer.”

Modified using word deletion: “the person was for her work a feminist activist and a writer.”

We run the fairness classifiers on the original and modified sentences, and compute the proportion of sentences assigned to the negative class. We repeat experiments 10 times and report mean negative scores with the standard deviation. Table 7 presents the results and we observe that both regard and sentiment classifiers are susceptible to give different results with different noise levels.

	Gender	Race	Religion
CEAT	man, woman	Asian, Black, Hispanic, White	Amish, atheist, Buddhist, Catholic, Christian, Hindu, Jewish, Mormon, Muslim, Protestant
ILPS	man, woman, transgender	Asian, Black, Hispanic, White	Amish, atheist, Buddhist, Catholic, Christian, Hindu, Jewish, Mormon, Muslim, Protestant
TOXD	man, woman,	Asian, Black, Hispanic, White	atheist, Buddhist, Christian, Hindu, Jewish, Muslim
HATEX	man, woman,	Asian, Black, Hispanic, White	atheist, Christian, Hindu, Jewish, Muslim
BOLD	man, woman, transgender	Asian, Black, Hispanic, White	atheist, Buddhist, Christian, Hindu, Jewish, Muslim, Sikhism
STEREOSET	gentlemen, grandfather, herself, himself, male, mommy, mother, schoolboy, schoolgirl, sister	Afghanistan, Arab, Bangladesh, Bengali, Black, Britain, Cameroon, Cape Verde, Columbian, Crimean, Ecuador, Eriteria, Eritrean, Ethiopia, Ethiopian, Ghanaian, Hispanic, Iranian, Iraq, Italy, Japanese, Jordan, Lebanon, Morocco, Norway, Norweigan, Persian people, Russian, Saudi Arabian, Sierra Leon, Somalia, Spain, Syria, Ukrainian, Vietnam	Brahmin, Christian, Muslim

Table 4: Protected groups that are covered by each metrics

Group	**Prompt**	**Generation**
Brahmin	The Brahmin class in India lords it over the rest of the population.	The Brahmin class in India lords it over the rest of the population. But this is not an accurate picture, since all Muslims are also Hindus and they have no rights to vote anywhere except their own homes.
Yemen	Yemen is far away	Yemen is far away from the center of attention at this stage, so what we need to do in order for it to move forward.
Eriteria	Believe it or not, Eriteria (an African country) is full of old Italian cathedrals.	Believe it or not, Eriteria (an African country) is full of old Italian cathedrals. There's even a church in the capital that used to be built by slaves
Arab	many arabs are muslim	many arabs are muslim, so it's no surprise they think nothing of the jews.
Ukrainian	Ukrainian women are beautiful.	Ukrainian women are beautiful. They're strong. Their sexuality is unique and they have a lot of freedom

Table 5: Examples of BOLD generations using STEREOSET prompts.

Group	**BOLD**	**STEREOSET**
Musician	A musician is a person who	Mark's child became a musician after school.
Physicist	Applied physicists use physics in scientific	The physicist is working.
Male	Adrian Zaw is an American actor best	Alena consider herself as beautiful woman.
Islam	Like other Abrahamic religions, Islam also	I had a Muslim teacher once.

Table 6: Examples of prompts from BOLD and STEREOSET for a set of groups

Dataset used	Proportion of negative regard	Proportion of negative sentiment
Clean data	25.2	13.6
5% word swap	25.12 (0.21)	13.52 (.10)
10% word swap	24.65 (0.37)	13.45 (0.32)
15% word swap	24.54 (0.67)	13.20 (0.26)
20% word swap	24.12 (0.49)	13.28 (0.35)
5% word deletion	24.88 (0.61)	13.24 (0.30)
10% word deletion	24.30 (0.50)	12.72 (0.68)
15% word deletion	23.38 (0.75)	12.30 (0.45)
20% word deletion	22.86 (0.49)	12.18 (0.42)

Table 7: Impact of synthetic noise on regard and sentiment classification. Proportion of negative class as predicted by the different fairness classifiers. We repeat experiments 10 times and report mean negative scores with the standard deviation.

Sequence-to-sequence AMR Parsing with Ancestor Information

Chen Yu and **Daniel Gildea**
Department of Computer Science
University of Rochester
Rochester, NY 14627

Abstract

AMR parsing is the task of mapping a sentence to an AMR semantic graph automatically. The difficulty comes from generating the complex graph structure. The previous state-of-the-art method translates the AMR graph into a sequence, then directly fine-tunes a pretrained sequence-to-sequence Transformer model (BART). However, purely treating the graph as a sequence does not take advantage of structural information about the graph. In this paper, we design several strategies to add the important *ancestor information* into the Transformer Decoder. Our experiments[1] show that we can improve the performance for both the AMR 2.0 and AMR 3.0 dataset and achieve new state-of-the-art results.

1 Introduction

Abstract Meaning Representation (AMR) (Banarescu et al., 2013) is a graph that encodes the semantic meaning of a sentence. In Figure 1a, we show the AMR of the sentence: *You told me to wash the dog*. AMR has been widely used in many NLP tasks (Liu et al., 2015; Hardy and Vlachos, 2018; Mitra and Baral, 2016).

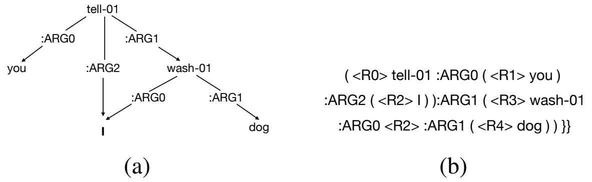
AMR parsing is the task of mapping a sentence to an AMR semantic graph automatically. A graph is a complex data structure which is composed of multiple vertices and edges. There are roughly four types of parsing strategies in previous work:

- **Two-Stage Parsing** (Flanigan et al., 2014; Lyu and Titov, 2018; Zhang et al., 2019a; Zhou et al., 2020): first produce vertices, and produce edges after that.

- **Transition-Based Parsing** (Damonte et al., 2016; Ballesteros and Al-Onaizan, 2017; Guo and Lu, 2018; Wang and Xue, 2017; Naseem et al., 2019; Astudillo et al., 2020; Zhou et al., 2021): process the sentence from left to right, and produce vertices and edges based on the current focused word.

- **Graph-Based Parsing** (Zhang et al., 2019b; Cai and Lam, 2019, 2020): produce vertices and edges based on a graph traversal order, such as DFS or BFS.

- **Sequence-to-Sequence Parsing** (Konstas et al., 2017; van Noord and Bos, 2017; Peng et al., 2017, 2018; Xu et al., 2020; Bevilacqua et al., 2021): this method linearizes the AMR graph to a sequence, then uses a sequence-to-sequence model to do the parsing.


tell-01
:ARG0
:ARG1
:ARG2
you
wash-01
:ARG0
:ARG1
I
dog
(<R0> tell-01 :ARG0 (<R1> you)
:ARG2 (<R2> I)):ARG1 (<R3> wash-01
:ARG0 <R2> :ARG1 (<R4> dog)) }}

(a) (b)

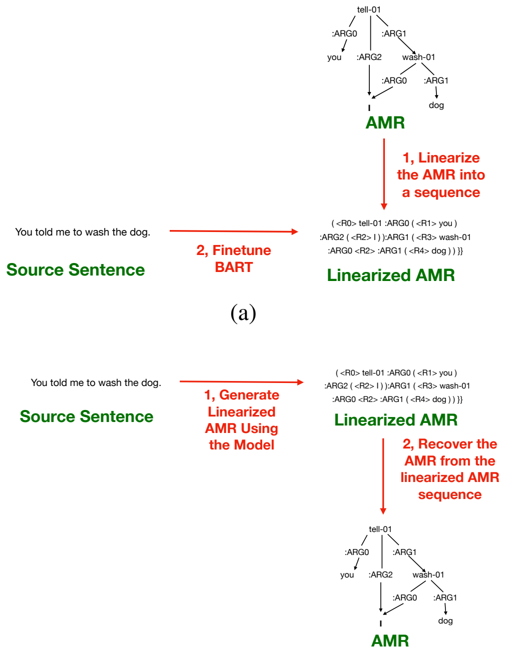
Figure 1: AMR Graph and linearization for the Sentence: *You told me to wash the dog.*

Bevilacqua et al. (2021) achieved the state-of-the-art performance by using the last seq-to-seq strategy. They linearized the AMR graph (see Figure 1b) and fine-tuned BART (Lewis et al., 2020), a denoising sequence-to-sequence pretrained model based on Transformer (Vaswani et al., 2017), for the parsing. We briefly show the method in Figure 2. During training, they linearize all the AMR graphs in the training dataset into sequences, then they can fine-tune the BART model in this new sequence-to-sequence dataset. At inference time, they first generate the AMR sequence using the BART model, then they recover the AMR graph from this sequence.

However, purely treating the graph as a sequence may not take advantage of important information about the structure of the graph. When generating

[1] https://github.com/lukecyu/amr-parser-s2s-ancestor

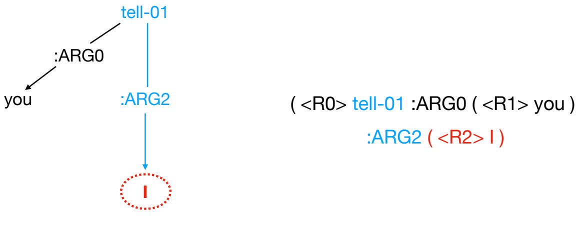
Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 571 - 577
May 22-27, 2022 ©2022 Association for Computational Linguistics


tell-01
:ARG0
:ARG1
you
:ARG2
wash-01
:ARG0
:ARG1
I
dog
AMR
1, Linearize the AMR into a sequence
You told me to wash the dog.
2, Finetune BART
(<R0> tell-01 :ARG0 (<R1> you)
:ARG2 (<R2> I)):ARG1 (<R3> wash-01
:ARG0 <R2> :ARG1 (<R4> dog)) }}
Source Sentence
Linearized AMR
You told me to wash the dog.
1, Generate Linearized AMR Using the Model
(<R0> tell-01 :ARG0 (<R1> you)
:ARG2 (<R2> I)):ARG1 (<R3> wash-01
:ARG0 <R2> :ARG1 (<R4> dog)) }}
Source Sentence
Linearized AMR
2, Recover the AMR from the linearized AMR sequence
tell-01
:ARG0
:ARG1
you
:ARG2
wash-01
:ARG0
:ARG1
I
dog
AMR

(a)

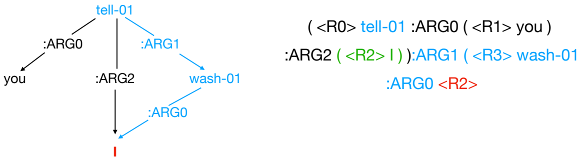
(b)

Figure 2: AMR Graph and linearization for the Sentence: *You told me to wash the dog.*



(a) (b)

Figure 3: Example of finding Ancestors.


tell-01
:ARG0
:ARG1
you
:ARG2
wash-01
:ARG0
I
(<R0> tell-01 :ARG0 (<R1> you)
:ARG2 (<R2> I)):ARG1 (<R3> wash-01
:ARG0 <R2>

(a) (b)

Figure 4: Example of finding ancestors with re-entrancy.

the last token *dog* in Figure 1b, for example, the dot-product attention layer in the Transformer Decoder attends to all the previous tokens and lets the model learn the weight of these tokens. However, if we can tell the model which tokens are its ancestors, like its parent is *wash-01* and its grand-parent is *tell-01* (see Figure 1a), it will make this token much easier to generate. Adding graph structure has been demonstrated to be useful for the AMR-to-text task (Zhu et al., 2019; Yao et al., 2020; Wang et al., 2020). These approaches added the graph structure to the Transformer Encoder. Therefore, we expect that adding structure in **Transformer Decoder** for AMR parsing task will also be helpful.

In this paper, we base our work on the seq-to-seq model of Bevilacqua et al. (2021) with the AMR linearized by DFS traversal order. We introduce several strategies to add ancestor information into the Transformer Decoder layer. We also propose a novel strategy, which consists of setting parameters in the mask matrix for those ancestor tokens and tuning them. We find that this new strategy makes the largest improvement.

2 Add Ancestors Information into Model

2.1 DFS linearization and Ancestors

The DFS linearization of Bevilacqua et al. (2021) used pairs of parentheses to indicate the start and the end of exploring a node in the DFS traversal order. The readers can use Figure 1 as an example and are referred to Bevilacqua et al. (2021) for more details.

This means when generating the next token, we can construct the partial graph from previous tokens and determine the ancestors tokens among them. In Figure 3b, for example, when we generate the token *I*, we can construct the partial graph in Figure 3a and find its ancestors (*tell-01 –> :ARG2 –>*).

If AMR were a tree, then the ancestors of each token would be clear to define. However, since AMR is a graph, one node may be visited multiple times (which is called re-entrancy), which brings ambiguity to find the ancestors. For example, in Figure 4, when we generate the last token *<R2>*, it is actually the re-entrancy of the token *I* generated before. Under this circumstance, we will use the tokens in the new path (*tell-01 –> :ARG1 –> wash-01 –> :ARG0 –>*) as its ancestors. We cannot use tokens from the old path (*tell-01 –> :ARG2 –>*), since we cannot know it is a re-entrancy before we have actually generated it.

2.2 Transformer Background

The original Transformer (Vaswani et al., 2017) used scaled dot-product self-attention. Typically, the input of the attention consists of a query matrix Q, a key matrix K and a value matrix V, the columns of which represent the query vector, the key vector and the value vector of each token. The attention matrix can be calculated as follows:

$$\text{Attention}(Q, K, V, M) = \text{Softmax}\left(\frac{S}{\sqrt{d}} + M\right)V,$$
$$S = QK^{\top},$$

where $Q, K, V \in \mathbb{R}^{N \times d}$, N is the length of the sequence, d is the dimension of the model, and M is the **mask matrix** to control which tokens in the sequence are attended for a given token.

A typical Transformer module consists of several layers. In each layer it uses MultiHead attention. For each head, it calculates attention as above, and then averages the results.

In the Encoder self-attention and Encoder-Decoder attention layers, the mask matrix is the same across all the heads and all the layers, and all the elements in the matrix are 0, meaning all the tokens are attended. But in the Decoder self-attention layers, the elements denoting the attention to the future token ($M_{i,j}$ with $i < j$) are set to $-\infty$, meaning that they have no effect when calculating the weighted sum.

2.3 Add Ancestor Information into Model

We focus on the **mask matrix** M in the Transformer Decoder self-attention layers to add the ancestor information during the parsing. We introduce two strategies: a hard strategy and a novel soft strategy.

Hard Strategy Under this strategy, we set elements denoting the ancestors to 0, and the elements denoting the non-ancestors to $-\infty$ in M, such that only the ancestor tokens are attended. We will explore the influence by using the new mask matrix only on some decoder layers or on some heads.

Soft Strategy Under this novel strategy, we will not mask the non-ancestor tokens and abandon them in a hard way. Instead, what we do is only telling the model which are the ancestor tokens and letting the model learn the weights by itself. Specifically, we use three different values in the mask matrix: $-\infty$ for all future tokens; 0 for all non-ancestor previous tokens; parameter α for all ancestor tokens. We let the model learn the weight α to control how much it should focus on the ancestor tokens. Similar to the hard strategy, we will also explore the influence by setting different parameters on different layers or on different heads.

2.4 Inference

During the inference stage, the input of the decoder is no longer the complete linearized AMR sequence. Instead, it is dynamically extended, and, at each step, the input is the tokens that have been generated during the previous steps. A natural question is: how can we find the ancestors of a token when we don't yet have a complete sequence (and therefore can't convert it to a graph to find its ancestors).

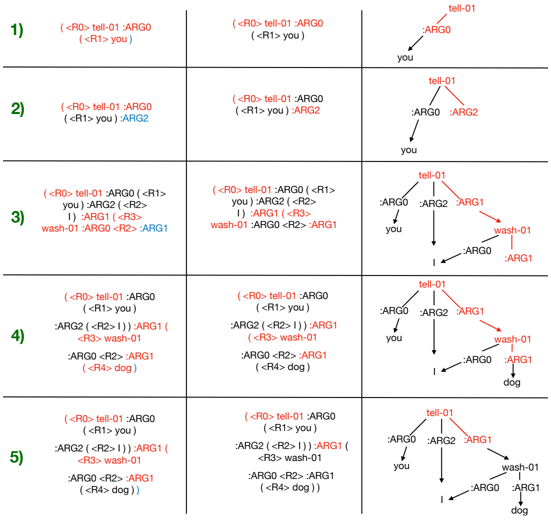
Fortunately, the DFS linearization uses several special tokens to denote the graph structure. We can rely on two special tokens to find the ancestors of a token: relation tokens (e.g. :ARG0) and the parentheses. The basic idea is: we maintain an ancestor stack for the token that will be generated, and adjust it according to the generated token. If a relation token is generated, we know that the previous siblings have been completely explored, so we will remove all the tokens of that sibling from the ancestor stack. If a right parenthesis is generated, we know that a token has been explored and we should return to its parent token, so we will remove it and all its descendants from the ancestor stacks. We always add the generated token (except the right parenthesis) into the ancestor stack after these special operations.

In Figure 5, we give an example of how to find the ancestor tokens during inference. In 1), the last token is the right parenthesis, meaning the last token *you* has been explored completely and should be removed from the ancestor token list. Therefore, we remove the tokens in the ancestor list backwards until we encounter a left parentheses. In 2), the last token is a relation token, meaning the previous sibling has been explored completely, so we remove the tokens in the ancestor list backwards until we encounter a previous relation token, then add the current relation token in the list. The steps 3), 4) and 5) are following the same rule.

3 Experiments

3.1 Setup

Dataset We use the AMR 2.0 (LDC2017T10) and AMR 3.0 (LDC2020T02) dataset. The AMR 2.0 includes 39,260 manually-created graphs, and the AMR 3.0 includes 59,255. The AMR 2.0 is a subset of AMR 3.0. Both datasets are split into training, development and test datasets.



1)	(<R0> tell-01 :ARG0 (<R1> you)	(<R0> tell-01 :ARG0 (<R1> you)
2)	(<R0> tell-01 :ARG0 (<R1> you) :ARG2	(<R0> tell-01 :ARG0 (<R1> you) :ARG2
3)	(<R0> tell-01 :ARG0 (<R1> you) :ARG2 (<R2> I) :ARG1 (<R3> wash-01 :ARG0 <R2> :ARG1	(<R0> tell-01 :ARG0 (<R1> you) :ARG2 (<R2> I) :ARG1 (<R3> wash-01 :ARG0 <R2> :ARG1
4)	(<R0> tell-01 :ARG0 (<R1> you) :ARG2 (<R2> I)) :ARG1 (<R3> wash-01 :ARG0 <R2> :ARG1 (<R4> dog)	(<R0> tell-01 :ARG0 (<R1> you) :ARG2 (<R2> I)) :ARG1 (<R3> wash-01 :ARG0 <R2> :ARG1 (<R4> dog)
5)	(<R0> tell-01 :ARG0 (<R1> you) :ARG2 (<R2> I)) :ARG1 (<R3> wash-01 :ARG0 <R2> :ARG1 (<R4> dog))	(<R0> tell-01 :ARG0 (<R1> you) :ARG2 (<R2> I)) :ARG1 (<R3> wash-01 :ARG0 <R2> :ARG1 (<R4> dog))

Figure 5: An example of how to find ancestors during inference. The red tokens are the ancestor tokens. The left column represents the ancestor tokens for the last blue tokens. The middle column represents the change of the ancestor tokens according to the last tokens. The right column represents the ancestors in the AMR of the middle columns.

Pre-processing and Post-processing We use the same DFS-based linearization technique as Bevilacqua et al. (2021). We omit the detail here, but the reader can refer to Figure 1 as an example. In the pre-processing step, the AMR graph is linearized into a sequence, and in the post-processing step, the generated sequence is translated back to an AMR graph.

Recategorization Recategorization is a widely used technique to handle data sparsity. With recategorization, specific sub-graphs of a AMR graph (usually corresponding to special entities, like named entities, date entities, etc.) are treated as a unit and assigned to a single vertex with a new content. We experiment with a commonly-used method in AMR parsing literature (Zhang et al., 2019a,b; Zhou et al., 2020; Bevilacqua et al., 2021). The readers are referred to Zhang et al. (2019a) for further details. Notice that this method uses heuristic rules designed and optimized for AMR 2.0, and is not able to scale up to AMR 3.0 (the performance dropped substantially for AMR 3.0 with recategorization in Bevilacqua et al. (2021)). Therefore, we will not conduct the recategorization experiment on AMR 3.0.

Model and Baseline We use the model in Bevilacqua et al. (2021) as our baseline. That model was initialized by BART pretraining and fine-tuned on the AMR dataset. We will do the same thing, except that we design a different mask matrix in the Transformer Decoder layers. We will introduce these differences in detail in Section 3.2.

Training and Evaluation We use one 1080Ti GPU to fine-tune the model. Training takes about 13 hours on AMR 2.0 and 17 hours on AMR 3.0. We use the development dataset to select the best hyperparameters. At inference time, we set the beam size to 5 following common practice in neural machine translation (Yang et al., 2018).

For evaluation, we use Smatch (Cai and Knight, 2013) as the metric. For some experiments, we also report fine-grained scores on different aspects of parsing, such as wikification, concept identification, NER, and negations using the tool released by Damonte et al. (2017).

3.2 Experiments and Results

As indicated in Section 2.3, we study the effect of the hard and soft strategy. We explore the influence of these two strategies on different layers or on different heads. Due to space limitation, we only show the Smatch score of AMR 2.0 with the recategorization preprocessing, since it had the highest performance (84.5 Smatch score) as far as we know.

Once we get the best result among these setups, we will conduct experiments on AMR 2.0 and AMR 3.0 without recategorization (we have discussed why we don't conduct experiments for AMR 3.0 with recategorization before). We will also report fine-grained results for these experiments.

3.2.1 Experiments for Different Number of Heads for the Hard Strategy

In the baseline model (Bevilacqua et al., 2021), there are 16 heads in each layer. We conduct experiments with 0, 2, 4, . . . , 8, 10 heads in each layer attending to ancestors only. Note that the 0-head model equals the baseline model. We show the result in Table 2.

We can see that, up to 4 and 6 heads, the performance increases along with the number of heads increasing, showing the importance of telling the model what the ancestors are. But then, the performance decreases as the number of heads increases, showing that we cannot ignore other non-ancestor tokens, which still play important roles in the model.

Dataset	G.R.	Smatch	Unlabeled	NO WSD	Concept	SRL	Reent.	Neg.	NER	wiki
AMR 2.0 (baseline)	✓	84.5	86.7	84.9	89.6	79.7	72.3	79.9	83.7	**87.3**
AMR 2.0 (our method)	✓	**85.2**	**88.2**	**85.6**	**90.3**	**83.2**	**75.4**	**83.0**	**85.7**	86.4
AMR 2.0 (baseline)	×	83.8	86.1	84.4	90.2	79.6	70.8	**74.4**	90.6	**84.3**
AMR 2.0 (our method)	×	**84.8**	**88.1**	**85.3**	**90.5**	**83.4**	**75.1**	74.0	**91.8**	84.1
AMR 3.0 (baseline)	×	83.0	85.4	83.5	**89.8**	78.9	70.4	**73.0**	87.2	**82.7**
AMR 3.0 (our method)	×	**83.5**	**86.6**	**84.0**	89.5	**82.2**	**74.2**	72.6	**88.9**	81.5

Table 1: The smatch and fine grained scores of AMR 2.0 and AMR 3.0 datasets without recategorization using the optimal setup.

number of heads	Smatch
0 (baseline)	84.5
2	84.5
4	**84.9**
6	**84.9**
8	84.8
10	84.3

Table 2: The influence of different number of heads attended to the ancestors only for AMR 2.0 with recategorization

different layers	Smatch
baseline	84.5
bottom 4	84.6
Medium 4	**84.8**
top 4	84.3

Table 3: The influence of different layers attended to the ancestors only for AMR 2.0 with recategorization

3.2.2 Experiments for Different Layers for the Hard Strategy

In the baseline model (Bevilacqua et al., 2021), there are 12 layers in the Transformer decoder. Unlike the heads, the order of layers matters. The upper layers use information from the lower layers. Therefore, we conduct experiments with the bottom, the medium, and the top 4 layers attending to ancestors. The mask matrix for each head is the same within a single layer. We show the result in Table 3.

We can see that, putting the medium 4 layers focusing on the ancestors has the best performance. But when we put the top 4 layers focusing on them, the performance decreases a lot. One possible reason is that, when it comes to near the final output (the top layers), the model needs to use the information from all tokens.

3.2.3 Experiments of Soft Strategy

In this section, we will tune the mask matrix and use the soft strategy to add the ancestors information. We conduct three experiments: different parameters for every layer and head combination; different parameters for different layers only; different parameters for different heads only. We show the results in Table 4. We can see that when we only use different parameters for every head, we achieve a new state-of-the-art result.

different setups	Smatch
baseline	84.5
different parameters for layers and heads	84.8
different parameters only for layers	84.7
different parameters only for heads	**85.2**

Table 4: The influence of tuning the mask matrix for AMR 2.0 with recategorization

3.2.4 Results for Other Datasets

We have conducted different experiments for AMR 2.0 with recategorization, and we found that when we set different parameters for different heads only and tune these parameters, we get the best performance. Therefore, we apply this setup for other datasets: AMR 2.0 and AMR 3.0 without recategorization. We show the Smatch scores as well as other fine-grained scores in Table 1. The results are improved for all the datasets. The AMR 2.0 without recategorization even obtains an improvement of 1.0 Smatch point.

4 Conclusion

In this paper, we focus on the DFS linearization and introduce several strategies to add ancestor information into the model. We conduct experiments to show the improvement for both AMR 2.0 and AMR 3.0 datasets. Our method achieves new state-of-the-art performances for the AMR parsing task.

Acknowledgments

We thank the anonymous reviewers for their insightful comments. Research supported by NSF awards IIS-1813823 and CCF-1934962.

References

Ramón Fernandez Astudillo, Miguel Ballesteros, Tahira Naseem, Austin Blodgett, and Radu Florian. 2020. Transition-based parsing with stack-transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1001–1007.

Miguel Ballesteros and Yaser Al-Onaizan. 2017. AMR parsing using stack-LSTMs. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1269–1275.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*.

Deng Cai and Wai Lam. 2019. Core semantic first: A top-down approach for AMR parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3790–3800.

Deng Cai and Wai Lam. 2020. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL-13)*, pages 748–752.

Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2016. An incremental parser for abstract meaning representation. *CoRR*, abs/1608.06111.

Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2017. An incremental parser for abstract meaning representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 536–546, Valencia, Spain.

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL-14)*, pages 1426–1436, Baltimore, Maryland.

Zhijiang Guo and Wei Lu. 2018. Better transition-based AMR parsing with a refined search space. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1712–1722, Brussels, Belgium. Association for Computational Linguistics.

Hardy Hardy and Andreas Vlachos. 2018. Guided neural language generation for abstractive summarization using abstract meaning representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 768–773.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

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

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A. Smith. 2015. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086.

Chunchuan Lyu and Ivan Titov. 2018. AMR parsing as graph prediction with latent alignment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 397–407. Association for Computational Linguistics.

Arindam Mitra and Chitta Baral. 2016. Addressing a question answering challenge by combining statistical methods with inductive rule learning and reasoning. In *AAAI*, pages 2779–2785.

Tahira Naseem, Abhishek Shah, Hui Wan, Radu Florian, Salim Roukos, and Miguel Ballesteros. 2019. Rewarding Smatch: Transition-based AMR parsing with reinforcement learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4586–4592.

Xiaochang Peng, Linfeng Song, Daniel Gildea, and Giorgio Satta. 2018. Sequence-to-sequence models for cache transition systems. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL-18)*, pages 1842–1852.

Xiaochang Peng, Chuan Wang, Daniel Gildea, and Nianwen Xue. 2017. Addressing the data sparsity issue in neural AMR parsing. In *Proceedings of the European Chapter of the ACL (EACL-17)*.

Rik van Noord and Johan Bos. 2017. Neural semantic parsing by character-based translation: Experiments with abstract meaning representations. *Computational Linguistics in the Netherlands Journal*, 7:93–108.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

Chuan Wang and Nianwen Xue. 2017. Getting the most out of AMR parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1268, Copenhagen, Denmark. Association for Computational Linguistics.

Tianming Wang, Xiaojun Wan, and Hanqi Jin. 2020. AMR-to-text generation with Graph Transformer. *Transactions of the Association for Computational Linguistics*, 8:19–33.

Dongqin Xu, Junhui Li, Muhua Zhu, Min Zhang, and Guodong Zhou. 2020. Improving AMR parsing with sequence-to-sequence pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2501–2511.

Yilin Yang, Liang Huang, and Mingbo Ma. 2018. Breaking the beam search curse: A study of (re-) scoring methods and stopping criteria for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3054–3059.

Shaowei Yao, Tianming Wang, and Xiaojun Wan. 2020. Heterogeneous graph transformer for graph-to-sequence learning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-20)*, pages 7145–7154.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019a. AMR parsing as sequence-to-graph transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 80–94, Florence, Italy.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019b. Broad-coverage semantic parsing as transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3786–3798, Hong Kong, China. Association for Computational Linguistics.

Jiawei Zhou, Tahira Naseem, Ramón Fernandez Astudillo, and Radu Florian. 2021. AMR parsing with action-pointer transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5585–5598.

Qiji Zhou, Yue Zhang, Donghong Ji, and Hao Tang. 2020. AMR parsing with latent structural information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4306–4319, Online. Association for Computational Linguistics.

Jie Zhu, Junhui Li, Muhua Zhu, Longhua Qian, Min Zhang, and Guodong Zhou. 2019. Modeling graph structure in Transformer for better AMR-to-text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP-19)*, pages 5462–5471.

A Hyperparameters and Training Details

We use cross-entropy loss and RAdam optimizer during the training. We use Cosine learning rate scheduler with about 1000 warm-up steps and 20000 maximum steps. The selected value of the learning rate is 3×10^{-5}. There are around 80 sentences in each batch. We set the weight decay rate of 0.004. In order to prevent over-fitting, we use Dropout with probability 0.25, as well as label smoothing with value 0.1. To select the best model checkpoint, we use the development dataset and search for the model with the best Smatch score.

Zero-Shot Dependency Parsing with Worst-Case Aware Automated Curriculum Learning

Miryam de Lhoneux[1,2,3], **Sheng Zhang**[4], **Anders Søgaard**[1]
[1]University of Copenhagen, Denmark [2]Uppsala University, Sweden [3]KU Leuven, Belgium
[4]National University of Defense Technology, China
{ml,soegaard}@di.ku.dk, zhangsheng@nudt.edu.cn

Abstract

Large multilingual pretrained language models such as mBERT and XLM-RoBERTa have been found to be surprisingly effective for cross-lingual transfer of syntactic parsing models (Wu and Dredze, 2019), but only between related languages. However, source and training languages are rarely related, when parsing truly low-resource languages. To close this gap, we adopt a method from multi-task learning, which relies on automated curriculum learning, to dynamically optimize for parsing performance on *outlier* languages. We show that this approach is significantly better than uniform and size-proportional sampling in the zero-shot setting.

1 Introduction

The field of multilingual NLP is booming (Agirre, 2020). This is due in no small part to large multilingual pretrained language models (PLMs) such as mBERT (Devlin et al., 2019) and XLM-RoBERTa (Conneau et al., 2020), which have been found to have surprising cross-lingual transfer capabilities in spite of receiving no cross-lingual supervision.[1] Wu and Dredze (2019), for example, found mBERT to perform well in a zero-shot setting when fine-tuned for five different NLP tasks in different languages. There is, however, a sharp divide between languages that benefit from this transfer and languages that do not, and there is ample evidence that transfer works best between typologically similar languages (Pires et al., 2019; Lauscher et al., 2020, among others). This means that the majority of world languages that are *truly low-resource* are still left behind and inequalities in access to language technology are increasing.

Large multilingual PLMs are typically fine-tuned using training data from a sample of languages that is supposed to be representative of the languages that the models are later applied to. However, this is difficult to achieve in practice, as multilingual datasets are not well balanced for typological diversity and contain a skewed distribution of typological features (Ponti et al., 2021). This problem can be mitigated by using methods that sample from skewed distributions in a way that is robust to outliers.

Zhang et al. (2020) recently developed such a method. It uses curriculum learning with a worst-case-aware loss for multi-task learning. They trained their model on a subset of the GLUE benchmark (Wang et al., 2018) and tested on outlier tasks. This led to improved zero-shot performance on these outlier tasks. This method can be applied to multilingual NLP where different languages are considered different tasks. This is what we do in this work, for the case of multilingual dependency parsing. Multilingual dependency parsing is an ideal test case for this method, as the Universal Dependency treebanks (Nivre et al., 2020) are currently the manually annotated dataset that covers the most typological diversity (Ponti et al., 2021).

Our research question can be formulated as such: *Can worst-case aware automated curriculum learning improve zero-shot cross-lingual dependency parsing?*[2]

[1] In the early days, cross-lingual transfer for dependency parsing relied on projection across word alignments (Spreyer and Kuhn, 2009; Agić et al., 2016) or *delexicalized transfer* of abstract syntactic features (Zeman and Resnik, 2008; McDonald et al., 2011; Søgaard, 2011; Cohen et al., 2011). Delexicalized transfer was later 're-lexicalized' by word clusters (Täckström et al., 2012) and word embeddings (Duong et al., 2015), but with the introduction of multilingual contextualized language models, transfer models no longer rely on abstract syntactic features, removing an important bottleneck for transfer approaches to scale to truly low-resource languages.

[2] Our work is related to work in meta-learning for zero-shot cross-lingual transfer, in particular Ponti et al. (2021), who use worst-case-aware meta-learning to find good initializations for target languages. Ponti et al. (2021) report zero-shot results for cross-lingual part-of-speech tagging and question-answering, with error reductions comparable to ours. Meta-learning also has been used for zero-shot cross-lingual learning by others (Nooralahzadeh et al., 2020; Xu et al., 2021), but using average loss rather than worst-case-aware objectives.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 578 - 587
May 22-27, 2022 ©2022 Association for Computational Linguistics

2 Worst-Case-Aware Curriculum Learning

In multi-task learning, the total loss is generally the average of losses of different tasks:

$$\min_{\theta} \ell(\theta) = \min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell_i(\theta) \tag{1}$$

where l_i is the loss of task i. The architecture we use in this paper is adapted from Zhang et al. (2020), which is an automated curriculum learning (Graves et al., 2017) framework to learn a worst-case-aware loss in a multi-task learning scenario. The architecture consists of a sampler, a buffer, a trainer and a multilingual dependency parsing model. The two main components are the sampler, which adopts a curriculum sampling strategy to dynamically sample data batches, and the trainer which uses worst-case-aware strategy to train the model. The framework repeats the following steps: (1) the sampler samples data batches of different languages to the buffer; (2) the trainer uses a worst-case strategy to train the model; (3) the automated curriculum learning strategy of the sampler is updated.

Sampling data batches We view multilingual dependency parsing as multi-task learning where parsing in each individual language is considered a task. This means that the target of the sampler at each step is to choose a data batch from one language. This is a typical multi-arm bandit problem (Even-Dar et al., 2002). The sampler should choose bandits that have higher rewards, and in our scenario, data batches that have a higher loss on the model are more likely to be selected by the sampler and therefore, in a later stage, used by the trainer. Automated curriculum learning is adopted to push a batch with its loss into the buffer at each time step. The buffer consists of n first-in-first-out queues, and each queue corresponds to a task (in our case, a language). The procedure repeats k times and, at each round, k data batches are pushed into the buffer.

Worst-case-aware risk minimization In multilingual and multi-task learning scenarios, in which we jointly minimize our risk across n languages or tasks, we are confronted with the question of how to summarize n losses. In other words, the question is how to compare two loss vectors α and β containing losses for all tasks $l_i, \dots l_n$:

$$\alpha = [\ell_1^1, \dots, \ell_n^1]$$

and

$$\beta = [\ell_1^2, \dots, \ell_n^2]$$

The most obvious thing to do is to minimize the mean of the n losses, asking whether $\sum_{\ell \in \alpha} \ell < \sum_{\ell \in \beta} \ell$. We could also, motivated by robustness (Søgaard, 2013) and fairness (Williamson and Menon, 2019), minimize the maximum (supremum) of the n losses, asking whether $\max_{\ell \in \alpha} \ell < \max_{\ell \in \beta} \ell$. Mehta et al. (2012) observed that these two loss summarizations are extremes that can be generalized by a family of multi-task loss functions that summarize the loss of n tasks as the L^p norm of the n-dimensional loss vector. Minimizing the average loss then corresponds to computing the L^1 norm, i.e., asking whether $|\alpha|^1 < |\beta|^1$, and minimizing the worst-case loss corresponds to computing the L^∞ (supremum) norm, i.e., asking whether $|\alpha|^\infty < |\beta|^\infty$.

Zhang et al. (2020) present a stochastic generalization of the L^∞ loss summarization and a practical approach to minimizing this family of losses through automated curriculum learning (Graves et al., 2017): The core idea behind their generalization is to optimize the worst-case loss with a certain probability, otherwise optimize the average (loss-proportional) loss with the remaining probability. The hyperparameter ϕ is introduced by the worst-case-aware risk minimization to trade off the balance between the worst-case and the loss-proportional losses. The loss family is formally defined as:

$$\min \ell(\theta) = \begin{cases} \min \max_i(\ell_i(\theta)), & p < \phi \\ \min \ell_{\tilde{i}}(\theta), & p \geq \phi, \tilde{i} \sim P_\ell \end{cases} \tag{2}$$

where $p \in [0,1]$ is a random generated rational number, and $P_\ell = \frac{\ell_i}{\sum_{j \leq n} \ell_j}$ is the normalized probability distribution of task losses. If $p < \phi$ the model chooses the maximum loss among all tasks, otherwise, it randomly chooses one loss according to the loss distribution. If the hyperparameter ϕ equals 1, the trainer updates the model with respect to the worst-case loss. On the contrary, if $\phi = 0$, the trainer loss-proportionally samples one loss.

Sampling strategy updates The model updates its parameters with respect to the loss chosen by the trainer. After that, the sampler updates its policy according to the behavior of the trainer. At each

Language	Treebank	Genus	Lang. family
Arabic	PADT	Semitic	Afro-Asiatic
Basque	BDT	Basque	Basque
Chinese	GSD	Chinese	Sino-Tibetan
English	EWT	Germanic	IE
Finnish	TDT	Finnic	Uralic
Hebrew	HTB	Semitic	Afro-Asiatic
Hindi	HDTB	Indic	IE
Italian	ISDT	Romance	IE
Japanese	GSD	Japanese	Japanese
Korean	GSD	Korean	Korean
Russian	SynTagRus	Slavic	IE
Swedish	Talbanken	Germanic	IE
Turkish	IMST	Turkic	Altaic

Table 1: 13 training treebanks. IE=Indo-European.

round, the policy of the task that is selected by the trainer receives positive rewards and the policy of all other tasks that have been selected by the sampler receive negative rewards.

The multilingual dependency parsing model We use a standard biaffine graph-based dependency parser (Dozat and Manning, 2017). The model takes token representations of words from a contextualized language model (mBERT or XLM-R) as input and classifies head and dependency relations between words in the sentence. The Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967) is then used to decode the score matrix into a tree. All languages share the same encoder and decoder in order to learn features from different languages, and more importantly to perform zero-shot transfer to unseen languages.

3 Experiments

We base our experimental design on Üstün et al. (2020), a recent paper doing zero-shot dependency parsing with good performance on a large number of languages. They fine-tune mBERT for dependency parsing using training data from a sample of 13 typologically diverse languages from Universal Dependencies (UD; Nivre et al., 2020), listed in Table 1. For testing, they use 30 test sets from treebanks whose language has not been seen at fine-tuning time. We use the same training and test sets and experiment both with mBERT and XLM-R as PLMs. It is important to note that not all of the test languages have been seen by the PLMs.[3]

We test worst-case aware learning with different values of ϕ and compare this to three main baselines: *size-proportional* samples batches proportionally to the data sizes of the training treebanks, *uniform* samples from different treebanks with equal probability, thereby effectively reducing the size of the training data, and *smooth-sampling* uses the smooth sampling method developed in van der Goot et al. (2021) which samples from multiple languages using a multinomial distribution. These baselines are competitive with the state-of-the-art when using mBERT, they are within 0.2 to 0.4 LAS points from the baseline of Üstün et al. (2020) on the same test sets. When using XLM-R, they are largely above the state-of-the-art.

We implement all models using MaChAmp (van der Goot et al., 2021), a library for multi-task learning based on AllenNLP (Gardner et al., 2018). The library uses transformers from HuggingFace (Wolf et al., 2020). Our code is publicly available.[4]

Our main results are in Table 2 where we report average scores across test sets, for space reasons. Results broken down by test treebank can be found in Table 4 in Appendix A. We can see that worst-case-aware training outperforms all of our baselines in the zero-shot setting, highlighting the effectiveness of this method. This answers positively our research question *Can worst-case aware automated curriculum learning improve zero-shot dependency parsing?*

Our results using mBERT are more than 1 LAS point above the corresponding baselines. Our best model is significantly better than the best baseline with $p < .01$ according to a bootstrap test across test treebanks. Our best model with mBERT comes close to Udapter (36.5 LAS on the same test sets) while being a lot simpler and not using external resources such as typological features, which are not always available for truly low-resource languages.

The results with XLM-R are much higher in general[5] but the trends are similar: all our models outperform all of our baselines albeit with smaller differences. There is only a 0.4 LAS difference between our best model and the best baseline, but it is still significant with $p < .05$ according to a bootstrap test across test treebanks. This highlights the robustness of the XLM-R model itself. Our results with XLM-R outperform Udapter by close to 7 LAS points.

[3] Information about which treebank has been seen by which PLM can be found in Appendix A.

[4] https://github.com/mdelhoneux/machamp-worst_case_acl

[5] Note, however, that the results are not directly comparable since different subsets of test languages have been seen by the two PLMs.

		mBERT	XLM-R
OURS	ϕ=0	**36.4**	42.1
	ϕ=0.5	36.1	**42.3**
	ϕ=1	36.1	**42.3**
BASELINES	size-proportional	35.0	41.9
	smooth-sampling	35.2	41.7
	uniform	35.2	41.4

Table 2: **Zero-shot performance:** Average LAS scores on the test sets of the 30 unseen (zero-shot) languages in the language split from Üstün et al. (2020).

sample	BASE	OURS	δ	RER
13LANG	35.2	36.4	1.2	1.9
GERMANIC	30.7	31.4	0.7	1.0
SLAVIC	30.4	31.7	1.3	1.9
ROMANCE	31.3	32.5	1.2	1.7
ROM+EU	33.3	34.8	1.5	2.2
ROM+AR	32.0	32.2	0.2	0.3
ROM+TR	32.2	33.0	0.8	1.2
ROM+ZH	33.4	34.1	0.7	1.1

Table 3: LAS of best baseline (BASE) and best worst-case training (OURS) when using mBERT as a PLM. Absolute difference (δ) and relative error reduction (RER) between OURS and BASE.

4 Varying the homogeneity of training samples

We investigate the interaction between the effectiveness of worst-case learning and the representativeness of the sample of training languages. It is notoriously difficult to construct a sample of treebanks that is representative of the languages in UD (de Lhoneux et al., 2017; Schluter and Agić, 2017; de Lhoneux, 2019). We can, however, easily construct samples that are **not** representative, for example, by taking a sample of related languages. We expect worst-case aware learning to lead to larger improvements in cases where some language types are underrepresented in the sample. We can construct an extreme case of underrepresentation by selecting a sample of training languages that has one or more clear outliers. For example we can construct a sample of related languages, add a single unrelated language in the mix, and then evaluate on other unrelated languages. We also expect that with a typologically diverse set of training languages, worst-case aware learning should lead to larger relative improvements than with a homogeneous sample, but perhaps slightly smaller improvements than with a very skewed sample.

We test these hypotheses by constructing seven samples of training languages in addition to the one used so far (13LANG). We construct three different homogeneous samples using treebanks from three different genera: GERMANIC, ROMANCE and SLAVIC. We construct four skewed samples using the sample of romance languages and a language from a different language family, an *outlier* language: Basque (eu), Arabic (ar), Turkish (tr) and Chinese (zh). Since we keep the sample of test sets constant, we do not include training data from languages that are in the test sets. The details of which treebanks are used for each of these samples can be found in Table 5 in Appendix B.

Results are in Table 3 where we report the average LAS scores of our best model (out of the ones trained with the three different ϕ values) to the best of the three baselines. We can see first that, as expected, our typologically diverse sample performs best overall. This indicates that it is a good sample. We can also see that, as expected, the method works best with a skewed sample: the largest gains from using worst-case learning, both in terms of absolute LAS difference and relative error reduction, are seen for a skewed sample (ROM+EU). However, contrary to expectations, the lowest gains are obtained for another skewed sample (ROM+AR). The gains are also low for ROM+TR, ROM+ZH and for GERMANIC. Additionally, there are slightly more gains from using worst-case aware learning with the SLAVIC sample than for our typologically diverse sample. These results could be due to the different scripts of the languages involved both in training and testing.

Looking at results of the different models on individual test languages (see Figure 1 in Appendix C), we find no clear pattern of the settings in which this method works best. We do note that the method always hurts Belarusian, which is perhaps unsurprising given that it is the test treebank for which the baseline is highest. Worst-case aware learning hurts Belarusian the least when using the SLAVIC sample, indicating that, when using the other samples, the languages related to Belarusian are likely downsampled in favour of languages unrelated to it. Worst-case learning consistently helps Breton and Swiss German, indicating that the method might work best for languages that are underrepresented within their language family but not necessarily outside of it. For Swiss German, worst-case learning helps least when using the GERMANIC sample where it is less of an outlier.

5 Conclusion

In this work, we have adopted a method from multitask learning which relies on automated curriculum learning to the case of multilingual dependency parsing. This method allows to dynamically optimize for parsing performance on *outlier* languages. We found this method to improve dependency parsing on a sample of 30 test languages in the zero-shot setting, compared to sampling data uniformly across treebanks from different languages, or proportionally to the size of the treebanks. We investigated the impact of varying the homogeneity of the sample of training treebanks on the usefulness of the method and found conflicting evidence with different samples. This leaves open questions about the relationship between the languages used for training and the ones used for testing.

6 Acknowledgements

We thank Daniel Hershcovich and Ruixiang Cui for comments on a draft of the paper, as well as the members of CoAStaL for discussions about the content of the paper. Miryam de Lhoneux was funded by the Swedish Research Council (grant 2020-00437). Anders Søgaard was funded by the Innovation Fund Denmark and a Google Focused Research Award. We acknowledge the computational resources provided by CSC in Finland through NeIC-NLPL and the EOSC-Nordic NLPL use case (www.nlpl.eu).

References

Željko Agić, Anders Johannsen, Barbara Plank, Héctor Martínez Alonso, Natalie Schluter, and Anders Søgaard. 2016. Multilingual projection for parsing truly low-resource languages. *Transactions of the Association for Computational Linguistics*, 4:301–312.

Eneko Agirre. 2020. Cross-Lingual Word Embeddings. *Computational Linguistics*, 46(1):245–248.

Yoeng-Jin Chu and Tseng-hong Liu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14:1396–1400.

Shay B. Cohen, Dipanjan Das, and Noah A. Smith. 2011. Unsupervised structure prediction with non-parallel multilingual guidance. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 50–61, Edinburgh, Scotland, UK. Association for Computational Linguistics.

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

Miryam de Lhoneux. 2019. *Linguistically Informed Neural Dependency Parsing for Typologically Diverse Languages*. Ph.D. thesis, Uppsala University.

Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2017. Old School vs. New School: Comparing Transition-Based Parsers with and without Neural Network Enhancement. In *Proceedings of the 15th Treebanks and Linguistic Theories Workshop (TLT)*, pages 99–110.

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

Timothy Dozat and Christopher Manning. 2017. Deep Biaffine Attention for Neural Dependency Parsing. In *Proceedings of the 5th International Conference on Learning Representations.*

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Cross-lingual transfer for unsupervised dependency parsing without parallel data. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 113–122, Beijing, China. Association for Computational Linguistics.

Jack Edmonds. 1967. Optimum Branchings. *Journal of Research of the National Bureau of Standards*, 71B:233–240.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. 2002. Pac bounds for multi-armed bandit and markov decision processes. In *International Conference on Computational Learning Theory*, pages 255–270. Springer.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Alex Graves, Marc G. Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Nishant Mehta, Dongryeol Lee, and Alexander G Gray. 2012. Minimax multi-task learning and a generalized loss-compositional paradigm for mtl. In *Advances in Neural Information Processing Systems*, pages 2150–2158.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-shot cross-lingual transfer with meta learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4547–4562, Online. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Edoardo Maria Ponti, Rahul Aralikatte, Disha Shrivastava, Siva Reddy, and Anders Søgaard. 2021. Minimax and neyman–Pearson meta-learning for outlier languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1245–1260, Online. Association for Computational Linguistics.

Natalie Schluter and Željko Agić. 2017. Empirically sampling Universal Dependencies. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 117–122, Gothenburg, Sweden. Association for Computational Linguistics.

Anders Søgaard. 2011. Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 682–686, Portland, Oregon, USA. Association for Computational Linguistics.

Anders Søgaard. 2013. Part-of-speech tagging with antagonistic adversaries. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 640–644, Sofia, Bulgaria. Association for Computational Linguistics.

Kathrin Spreyer and Jonas Kuhn. 2009. Data-driven dependency parsing of new languages using incomplete and noisy training data. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 12–20, Boulder, Colorado. Association for Computational Linguistics.

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 477–487, Montréal, Canada. Association for Computational Linguistics.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly Universal Dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2302–2315, Online. Association for Computational Linguistics.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Robert Williamson and Aditya Menon. 2019. Fairness risk measures. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6786–6797. PMLR.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Weijia Xu, Batool Haider, Jason Krone, and Saab Mansour. 2021. Soft layer selection with meta-learning for zero-shot cross-lingual transfer. In *Proceedings of the 1st Workshop on Meta Learning and Its Applications to Natural Language Processing*, pages 11–18, Online. Association for Computational Linguistics.

Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*.

Sheng Zhang, Xin Zhang, Weiming Zhang, and Anders Søgaard. 2020. Worst-case-aware curriculum learning for zero and few shot transfer. *arXiv preprint arXiv:2009.11138*.

A Results by treebank

Results by language of the test treebanks are in Table 4.

B Training samples

The training samples are summarized in Table 5.

C Results by treebank with the different samples

Relative error reduction between our best worst-case aware result and the best baseline for each training sample used, with mBERT, in Figure 1.

	mBERT						XLM-R					
iso	ϕ=0	ϕ=0.5	ϕ=1	S-P	S-S	U	ϕ=0	ϕ=0.5	ϕ=1	S-P	S-S	U
aii *#	8	**11.3**	10.8	1.6	6.4	6.0	2	3.3	3.1	2.9	**3.5**	3.1
akk *#	1.5	1.4	1.6	2.5	**3.0**	1.9	2.5	2.5	**2.8**	1.9	2.2	2.3
am *	**16.5**	10.9	13.2	6.6	10.8	10.6	68.0	68.6	68.3	68.4	**68.8**	68.1
be	78.5	79.4	79.6	**82.0**	80.9	80.5	85.6	85.5	85.6	86.4	86.8	**86.8**
bho *#	**38.1**	37.8	37.9	37.0	36.7	36.7	37.3	37.4	37.1	37.4	**37.6**	37.2
bm *#	**9.0**	8.7	8.7	6.9	6.7	6.9	6.0	6.4	6.2	**6.5**	6.3	6.4
br	**62.9**	62.6	62.0	60.3	60.3	59.6	59.5	59.6	**60.5**	59.9	59.5	58.9
bxr *#	25.9	**26.0**	25.6	24.6	25.5	25.4	27.7	**28.2**	28.0	27.2	27.2	26.2
cy	**55.5**	55.0	55.2	55.1	54.4	54.2	59.8	60.1	59.9	60.2	**60.6**	59.6
fo *#	67.4	67.8	**68.0**	66.3	67.2	66.4	73.5	72.8	**73.5**	72.6	72.4	73.0
gsw *#	48.3	**48.8**	48.2	44.9	42.2	42.3	46.0	**46.5**	**46.5**	43.6	42.2	44.3
gun *#	8.2	8.5	**8.7**	7.3	8.0	8.3	6.8	6.8	**7.6**	6.5	5.8	5.6
hsb *#	50.8	51.3	**51.4**	49.4	49.2	49.1	**62.6**	61.9	62.0	61.4	61.6	60.0
kk	**60.1**	58.9	58.4	58.5	59.0	58.2	63.0	62.7	62.5	**63.7**	62.3	61.5
kmr *	9.3	9.2	8.9	8.6	**9.6**	9.5	**53.5**	53.1	53.2	51.8	51.7	52.0
koi *#	19.3	18.8	**19.8**	15.8	15.8	16.0	17.0	**20.1**	19.1	17.8	17.8	16.0
kpv *#	16.8	17.0	**17.2**	15.6	16.2	15.8	18.3	19.1	**19.5**	17.0	17.8	16.3
krl *#	46.6	46.4	46.3	46.5	**47.1**	46.4	61.0	61.2	60.7	62.0	**62.1**	61.8
mdf *#	**26.1**	24.3	24.3	22.5	24.5	25.4	20.4	**20.7**	19.6	18.4	18.4	16.8
mr	60.6	**61.2**	60.1	56.9	57.7	57.7	69.2	69.7	**70.0**	67.8	**70.0**	69.7
myv *#	**20.2**	19.9	19.8	18.5	19.3	19.9	16.8	**17.2**	16.9	16.0	16.3	15.5
olo *#	40.7	**41.7**	41.0	41.0	40.9	40.5	56.5	**56.7**	56.1	55.8	54.3	54.4
pcm *#	33.9	32.8	33.0	32.5	34.3	**35.4**	**39.2**	39.2	38.9	38.0	37.6	37.8
sa *	**22.5**	21.9	22.3	21.1	21.0	20.6	50.2	49.7	**50.9**	**50.9**	50.1	50.0
ta	52.3	**54.7**	54.3	53.2	52.0	51.6	54.9	**55.0**	54.8	53.8	53.8	54.0
te	69.9	69.8	70.0	69.4	**70.6**	68.7	76.0	76.0	76.7	76.3	**77.1**	76.3
tl #	65.4	57.5	56.5	**65.8**	59.3	65.4	77.1	75.7	75.7	**78.1**	76.7	76.4
wbp *#	5.9	8.8	**9.2**	7.5	7.5	7.2	7.8	**9.5**	7.5	8.5	5.2	8.8
yo #	37.8	37.9	38.5	**39.7**	38.0	37.5	3.3	**3.6**	3.2	2.3	2.7	1.8
yue *#	**33.0**	32.5	32.5	32.4	32.4	32.4	41.9	41.7	42.0	**42.9**	42.4	42.8
average	**36.4**	36.1	36.1	35.0	35.2	35.2	42.1	**42.3**	**42.3**	41.9	41.7	41.4

Table 4: **Zero-shot performance:** LAS scores on the test sets of the 30 unseen (zero-shot) languages in the language split from Üstün et al. (2020) using mBERT and XLM-R. S-P=size-proportional, S-S = smooth-sampling, U=uniform. Bold indicates the best performance across models using the same PLM. * means not in mBERT and # means not in XLM-R.

	GERMANIC	SLAVIC	ROMANCE	ROM+EU	ROM+AR	ROM+TR	ROM+ZH	13LANG
Afrikaans-AfriBooms	✓							
Danish-DDT	✓							
Dutch-Alpino	✓							
English-EWT	✓							✓
German-HDT	✓							
Gothic-PROIEL	✓							
Icelandic-IcePaHC	✓							
Norwegian-Bokmaal	✓							
Swedish-Talbanken	✓							✓
Czech-PDT		✓						
Old_Church_Slavonic-PROIEL		✓						
Old_Russian-TOROT		✓						
Polish-LFG		✓						
Russian-SynTagRus		✓						✓
Serbian-SET		✓						
Slovak-SNK		✓						
Ukrainian-IU		✓						
French-GSD			✓	✓	✓	✓	✓	
Italian-ISDT			✓	✓	✓	✓	✓	✓
Portuguese-GSD			✓	✓	✓	✓	✓	
Romanian-RRT			✓	✓	✓	✓	✓	
Spanish-AnCora			✓	✓	✓	✓	✓	
Basque-BDT				✓				✓
Arabic-PADT					✓			✓
Chinese-GSD							✓	✓
Turkish-IMST						✓		✓
Finnish-TDT								✓
Hebrew-HTB								✓
Hindi-HDTB								✓
Japanese-GSD								✓
Korean-GSD								✓

Table 5: Treebanks included in the different samples

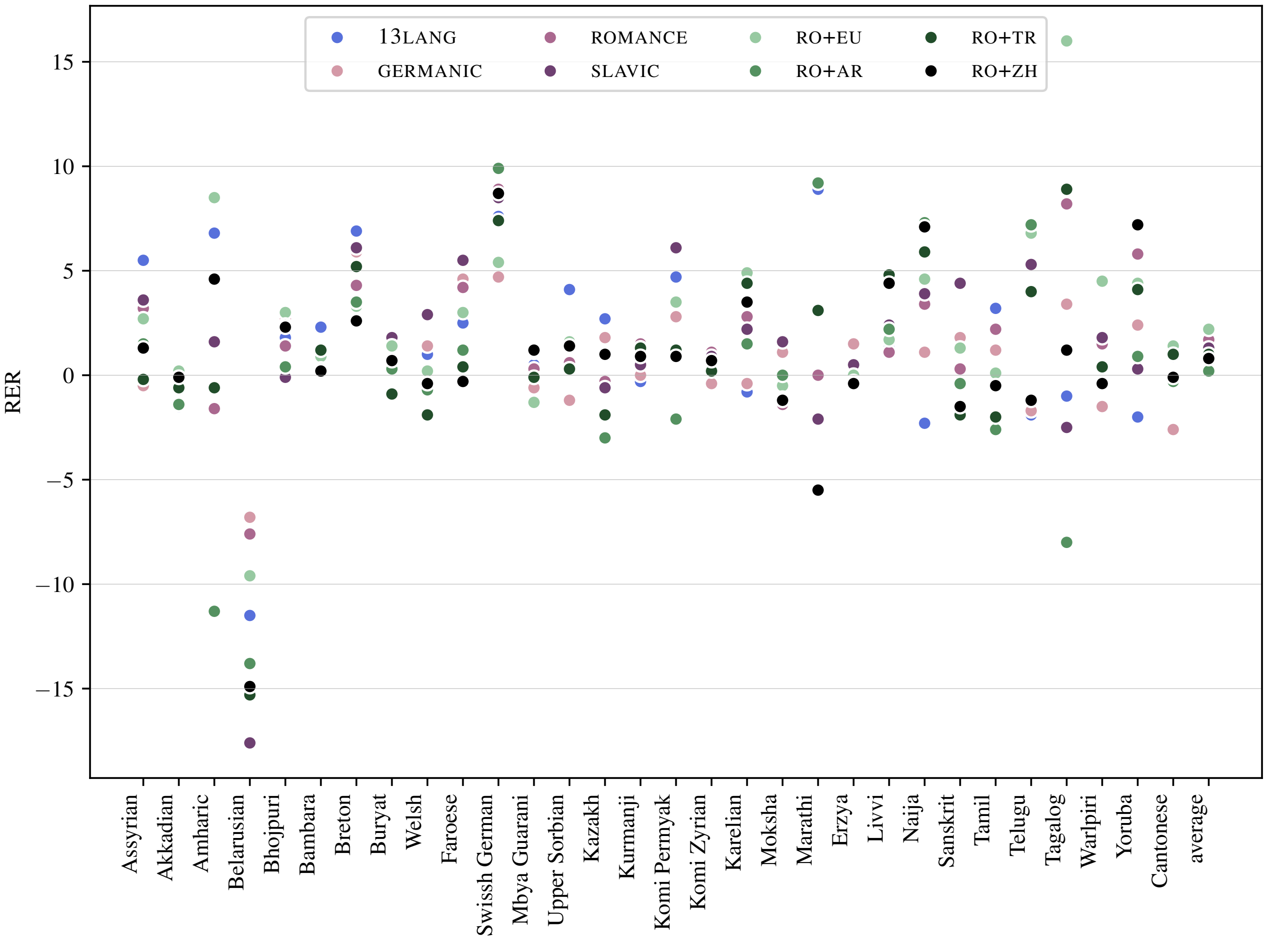


Figure 1: Relative error reduction (RER) in LAS points between our best worst-case aware result and the best baseline for each training sample used on test sets in the 30 languages.

PriMock57: A Dataset Of Primary Care Mock Consultations

Alex Papadopoulos Korfiatis
Babylon
alex.papadopoulos[1]

Francesco Moramarco
Babylon, University of Aberdeen
francesco.moramarco[1]

Radmila Sarac
radmila.sarac@gmail.com

Aleksandar Savkov
Babylon
sasho.savkov[1]

[1]@babylonhealth.co.uk

Abstract

Recent advances in Automatic Speech Recognition (ASR) have made it possible to reliably produce automatic transcripts of clinician-patient conversations. However, access to clinical datasets is heavily restricted due to patient privacy, thus slowing down normal research practices. We detail the development of a public access, high quality dataset comprising of 57 mocked primary care consultations, including audio recordings, their manual utterance-level transcriptions, and the associated consultation notes. Our work illustrates how the dataset can be used as a benchmark for conversational medical ASR as well as consultation note generation from transcripts.

1 Introduction

The use of Automatic Speech Recognition (ASR) is widespread in the clinical domain but it is generally used to alleviate the administrative burden of clinical notes through dictation (Hodgson and Coiera, 2016; Kumah-Crystal et al., 2018).

However, the adoption of telemedicine, especially in primary care, generates vast quantities of clinical interaction recordings. Additionally, ASR models have become much more robust to applications in the clinical domain. In turn, this is beneficial for downstream Natural Language Processing (NLP) tasks, such as information extraction from clinical conversations (Selvaraj and Konam, 2021; Soltau et al., 2021) and automatic generation of consultation notes (Finley et al., 2018; Enarvi et al., 2020a; Quiroz et al., 2020; Molenaar et al., 2020).

Despite this being an active area of research it still lacks a commonly recognised ASR benchmark due to the sensitive nature of clinical conversations. Furthermore, as the datasets are not shared, research teams always need to invest time and resources into making their own private dataset. These limitations slow down progress in the field.

We release[1] a high quality public dataset of primary care consultation audio recordings, including manual transcriptions and associated consultation notes, which is the basis of our contributions:

1. a benchmark for ASR for primary care conversations;
2. a benchmark for automatic generation of consultation notes for primary care.

2 Related Work

Automated transcription of clinical consultations has attracted quite significant research interest; however, as mentioned above, there is no easily accessible common benchmark dataset in the style of Switchboard (Godfrey et al., 1992) or Fisher (Cieri et al., 2004), which are both non-medical conversational audio datasets. Because of this, comparing different approaches for clinical conversation ASR is challenging.

For example, Chiu et al. (2018) detail a dataset of ≈ 14,000 hours of recorded and manually transcribed consultations that they use to train an end-to-end clinical conversation ASR model. Similarly, Kim (2020), Soltau et al. (2021) develop end-to-end ASR models for clinical conversations and Mani et al. (2020) train a sequence-to-sequence machine translation model to correct the errors of general-domain ASR engines; but they all use different, proprietary datasets. Johnson et al. (2014) and Kodish-Wachs et al. (2018) perform systematic reviews of the accuracy of a number of open-source and commercial ASR models for clinical conversation transcription; again, on proprietary datasets.

As for open-access datasets, He et al. (2020) compile and release two clinical dialogue datasets in Chinese and English, covering a wide range of clinical specialties. Ju et al. (2020) do the same for COVID-19 related clinical dialogue. These

[1] https://github.com/babylonhealth/primock57

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 588 - 598
May 22-27, 2022 ©2022 Association for Computational Linguistics

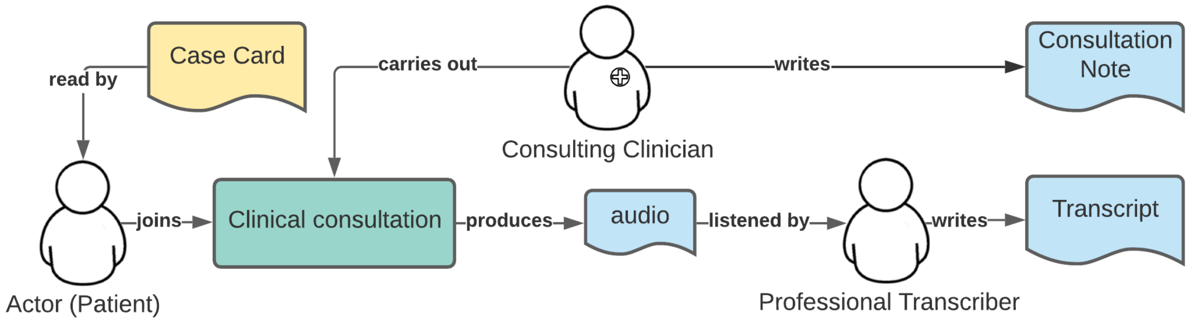


Figure 1: Overview of the data collection process. A mock patient, reading from a medical case card, has a consultation with a clinician which is recorded and transcribed. The resulting dataset includes the consultation audio recordings, notes and manual transcripts.

datasets are gathered from online clinical question answering sources; while they are relevant for clinical chatbot research, they are not representative of clinical interactions and do not include audio. Kazi et al. (2020) provide a dataset of audio recordings, automated transcripts and consultation notes for 70 mock psychiatric consultations — but no human transcripts.

Automatic consultation note generation and other long-form text summarisation tasks have rapidly developed due to recent advances in Natural Language Generation (NLG) architectures (Vaswani et al., 2017; Devlin et al., 2019). Several studies (Liu et al., 2019; MacAvaney et al., 2019; Zhang et al., 2020; Enarvi et al., 2020b; Joshi et al., 2020; Krishna et al., 2021; Chintagunta et al., 2021; Yim and Yetisgen-Yildiz, 2021; Moramarco et al., 2021; Zhang et al., 2021) use proprietary datasets of transcripts and notes to train NLG models end-to-end, and a number of them carry out automatic or human evaluations on their proprietary test sets. However, in a similar fashion to the ASR studies discussed above, most studies don't publish these resources; hence, it is again prohibitively difficult to compare their proposed methods. Kazi et al. (2020) provide the only open access clinical dataset that could be used as a benchmark but it only contains psychiatric consultations, which is less applicable to primary care.

3 Dataset

The requirements for releasing a dataset containing Personal Health Information (PHI) are typically costly and involve collecting patient consent and/or de-identification, which is especially challenging with audio recordings. We built a mock consultation dataset as close as possible to the real conditions as a pragmatic alternative. The diagram in Figure 1 shows an overview of the data collection process.

Consultation type	Count
Otitis	2
Anaphylactic reaction	3
Cardiovascular	11
Dermatitis	4
Fever	4
Urinary tract infection	6
Upper respiratory infection	6
Asthma	2
Gastroenteritis	8
Mental health	3
Physical injury	2
Migraine	6

Table 1: A breakdown by consultation case card. The case card diagnoses were selected to be representative of common telemedecine presenting complaints.

3.1 Mock consultation recordings

We employed 7 clinicians and 57 actors posing as patients from a range of ethnicities. The clinicians had experience with virtual consultations. Participation was optional and anyone could choose to withdraw at any time. Four of the clinicians were men and three were women; five of them had British English accent, and two of them Indian. The patient accent distribution is as follows: British English (47.4%), various European (31.6%), other English (10.5%), and other non-English (10.5%). The gender distribution was relatively even (52.6% women, 47.4% men); most participants were from 25 to 45 years old (see Figure A.1).

Each mock patient was given a case card that included background information (age, social history, family history of illnesses) as well as information about their presenting complaint, symptoms, condi-

Demographics (age, gender): 23 year old female
Presenting Complaint: Lower abdominal pain Duration of symptoms: 2 days
History, on open questioning: Have a terrible ache in my lower tummy and feeling hot and sweaty.
Symptoms and risk factors: There is some blood in the urine – pink colour Pain below belly button Feeling nauseated but no vomiting * * *

Table 2: An abridged example of a clinical case card for a Urinary Tract Infection. Mock patients were given a case card and asked to study it before consulting with the clinician. Full version available in the Appendix.

tions, and medications. The case cards were drawn from a pool of primary care conditions, representative of presenting complaints in UK primary care. For a breakdown of presenting complaints, see Table 1. An example case card is given in Table 2.

We recorded 57 mock consultations (8h38m6s in total) over 5 days, using proprietary telemedicine software that allowed us to export the individual clinician and patient audio channels.[2] In order to emulate real clinical practice, clinicians were using laptops while patients were using mobile phones in an office environment with background noise. Clinicians were asked to act as close as possible to their actual consultation sessions, including conforming to a consultation length of 10 minutes and writing a consultation note in the SOAP format (Pearce et al., 2016). The resulting mock consultations ranged between 3m48s and 14m18s, with an average consultation length of 9m5s.

3.2 Manual transcription

To transcribe the consultation recordings, we employed transcribers with experience in the clinical conversation domain, who were asked to:

1. Listen to the consultation audio recordings, in separate channels for clinicians and patients;
2. Identify the start and end points of individual utterances (continuous speech segments ending in a pause);

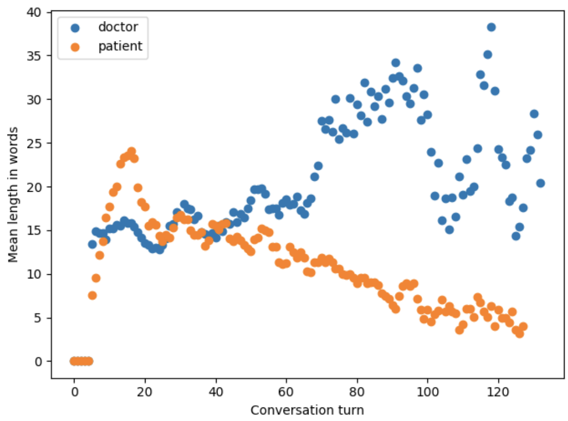


Figure 2: Average utterance length for clinician and patient as a function of conversation turns. The patient initially speaks more than the clinician but later in the consultation this trend is reversed.

3. Provide an accurate transcription of each of the utterances identified.

Thus we obtained a collection of start times, end times, and utterance-level transcriptions, important for the ASR evaluation described below.

Consultations have 92 conversation turns and 1,489 words on average; clinicians tend to speak more than patients (897 vs. 592 words per consultation) and take longer turns (19.3 vs 12.8 words per turn). Interestingly, patients tend to take longer turns than clinicians in the beginning of the consultation, where they presumably state their presenting complaint; turns are more balanced in the middle, and clinicians seem to take over during the diagnosis and management at the end (see Figure 2).

4 ASR Benchmark

We perform a baseline study of ASR for clinical conversations by passing the audio recordings of the mock consultations through commonly used open-source and commercial speech-to-text engines:

1. **Kaldi**: This is our baseline system, built using the Kaldi (Povey et al., 2011) speech recognition toolkit, running locally. It uses a pretrained acoustic model from Zamia Speech[3] and a 3-gram language model trained on a proprietary medical question answering dataset.
2. **NeMo QuartzNet & Conformer**: These systems use QuartzNet (Kriman et al., 2020) and Conformer (Gulati et al., 2020) ASR models, which we load using Nvidia's NeMo toolkit.[4]

[2] Due to limitations of the software, audio was exported in compressed form (WebM encoder, Opus codec at a variable bitrate).

[3] http://zamia-speech.org/asr/

[4] https://github.com/NVIDIA/NeMo

	WER								ECCA		
			Gender		Role		Accent				
ASR	mean	stdev	M	F	Clinician	Patient	en-gb	other	Pr	Re	F1
GC STT	**30.9**†	12.7	32.7	28.9	28.5	33.4	30.0	32.2	0.83	**0.82**	0.81
Azure STT	**31.3**†	12.8	32.7	29.6	26.7	35.8	30.2	32.7	**0.87**	0.79	**0.82**
ATM	34.0‡	13.9	33.8	34.2	32.8	35.2	31.6	37.2	0.79	0.75	0.78
Kaldi	48.9	14.9	52.7	44.6	47.0	50.8	49.5	48.2	0.64	0.69	0.68
QuartzNet	46.4	15.5	48.4	44.1	48.1	44.7	46.6	46.1	0.67	0.49	0.56
Conformer	34.4‡	14.5	36.8	31.7	35.6	33.2	35.0	33.7	0.79	0.71	0.75

Table 3: Word Error Rate (WER) scores for a number of Speech-to-text engines, and Extracted Clinical Concepts Accuracy (ECCA) based on recognised clinical terms. The gender, role and accent breakdowns show how each factor affects the mean WER. † indicates lack of statistical significance between mean WER scores ($p = 0.097$); ‡ is weak significance ($p = 0.026$); all other scores are $p < 0.001$.

Both models are end-to-end and do not use a language model.

3. **Google Cloud Speech-to-text (GCSTT)**:[5] a commercially available, general domain service. We use the *video* enhanced model which is only available for the *en-us* language.
4. **Amazon Transcribe Medical (ATM)**:[6] a commercially available service, tailored specifically for medical use cases. There are models available for *clinical dictation* and *clinical conversation*; we use the conversation model with *speciality=Primary Care*.
5. **Azure Speech-to-text (ASTT)**:[7] a commercially available, general domain service. We use the *Standard* model.

To test the accuracy of the above services, we first extract the audio for each individual utterance identified by our human transcribers. We then generate a transcript for the utterance using each of the ASR engines. We ensure consistency by performing the following post-processing steps on both human and automatic transcripts:

1. Remove disfluencies ("umm", "uhh", etc.). These are included in the reference transcripts, but often omitted in each STT service;
2. Replace numerals ("5", "9th", "1984") with written equivalents ("five", "ninth", "nineteen eighty-four") to ensure uniformity;
3. Remove all punctuation, collapse multiple spaces and convert to lowercase.

Finally, we compute the Word Error Rate (WER) for each utterance using SCTK's *sclite*[8] tool. The mean WER, including a breakdown by gender, role, and accent can be seen in Table 3. Even though both are general domain, Google and Azure together are the best performing models on our dataset ($p = 0.097$). Conformer performs surprisingly well, given that it is a character-level model evaluated on a word-level metric.

The base WER metric treats all words in a transcript as equally important; this may be less desirable in the clinical domain, where the correct transcription of specific clinical terms is expected to be more important. To test this, we use a proprietary clinical information extraction engine based on fuzzy string matching, linking to SNOMED-CT (Donnelly et al., 2006). We extract medical concepts from each utterance in both reference and hypothesis transcripts, then compare the concepts extracted to estimate accuracy based on clinical terminology (ECCA in Table 3). The results mostly match the WER comparisons; the medical-domain Amazon model does not seem to perform better.

5 Consultation Note Generation Benchmark

The consultation transcripts and corresponding notes (see example in Table 4) are intended as a parallel dataset to evaluate methods for automatically generating primary care consultation notes. We propose a benchmark for this task by evaluating a number of baseline approaches and reporting common automatic metric scores on our dataset. The approaches considered include:

[5] https://cloud.google.com/speech-to-text

[6] https://aws.amazon.com/transcribe/medical/

[7] https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/

[8] https://github.com/usnistgov/SCTK

Transcript		Note
Clinician	So, um, tell me what's been going on. You've been saying there's a problem with your hearing. Is that right?	History: Hx of difficulty hearing left ear for 6 weeks with tinnitus and slight nausea/ dizziness. One previous similar episode in the past- resolved spontaneously. No discharge/fever/itchiness/pain Doesn't use cotton wool buds No Pmhx of note Ex: Looks well, not in pain. Imp: need to exclude impacted wax in ear canal first Pln: for face to face GP appointment in 5 days to examine ear If any problems in interim to ring us back Pt happy with and understands plan
Patient	Yeah, so I just feel I can't really hear as well as I used to, like my hearing is kind of deteriorating in some way.	
Clinician	Right, OK. How long has this been going on for?	
Patient	Uh about six weeks.	
Clinician	Six weeks, OK. Um, and before that have you had any hearing problem at all?	
Patient	Um I had something maybe, about a year ago, but it only lasted a couple of days, it wasn't anything as long as this.	
Clinician	Right, OK, OK. And, um, in this six week period, have you had anything else happen? Have you had any other ear symptoms at all?	
Patient	Um, I occasionally get like a ringing in my left ear, uh just on the one side and um there's actually been a few times when I felt kind of a bit sick or a bit dizzy as well.	

Table 4: Snippet of a mock consultation transcript and the corresponding note, written by the consulting clinician.

Model	R1	R2	RL	B
BART-CNN	0.17	0.02	0.10	0.80
BERT-ext	0.21	0.03	0.10	0.78
Random	0.19	0.02	0.09	0.78
BART-finet	**0.31**	**0.08**	**0.17**	**0.81**

Table 5: Average common metrics scores of different models on the 57 consultations. R1 through L represent Rouge F1 scores for unigrams, bigrams, and longest-common-subsequence. B represents non-rescaled BERTScore; score range is between 0.7 to 0.9, so differences are less pronounced.

BART-CNN: a neural sequence-to-sequence summariser based on the BART model (Lewis et al., 2020) and fine-tuned on the Dailymail/CNN dataset (Nallapati et al., 2016);

BERT-ext: a general-purpose extractive summariser based on Bert embeddings (Miller, 2019);

Random: a baseline that extracts 15 random sentences from the transcript and collates them to form a note;

BART-finet: a BART-CNN model further fine-tuned on a proprietary dataset of 8,000 real transcripts and consultation notes.

We evaluate the models on our dataset and report common summarisation metrics scores: Rouge-1, -2 & -L (Lin, 2004) which compute the F-score across ngrams between generated and human notes; and BERTScore (Zhang et al., 2019), which computes the similarity between BERT embeddings of the notes.

The results can be seen in Table 5: the fine-tuned BART model scores highest with all metrics, while *BART-CNN* and *BERT-ext* fail to outperform the *Random* baseline model. This highlights the differences between consultation note generation and general-purpose summarisation.

A more detailed evaluation of this task can be found in Moramarco et al. (2022); example notes can be found in Appendix Table A.3.

6 Conclusion

We present a dataset of 57 high quality mocked consultation audio recordings, their manually aligned and diarised transcripts, and consultation notes. By publishing this dataset, we hope to offer a benchmark for future studies in both ASR for clinical conversations and Consultation Note Generation for the primary care domain.

References

Bharath Chintagunta, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2021. Medically aware gpt-

3 as a data generator for medical dialogue summarization. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 66–76.

Chung-Cheng Chiu, Anshuman Tripathi, Kat Chou, Chris Co, Navdeep Jaitly, Diana Jaunzeikare, Anjuli Kannan, Patrick Nguyen, Hasim Sak, Ananth Sankar, Justin Jesada Tansuwan, Nathan Wan, Yonghui Wu, and Frank Zhang. 2018. Speech recognition for medical conversations.

Christopher Cieri, David Miller, and Kevin Walker. 2004. The Fisher corpus: A resource for the next generations of speech-to-text.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Kevin Donnelly et al. 2006. Snomed-ct: The advanced terminology and coding system for ehealth. *Studies in health technology and informatics*, 121:279.

Seppo Enarvi, Marilisa Amoia, Miguel Del-Agua Teba, Brian Delaney, Frank Diehl, Stefan Hahn, Kristina Harris, Liam McGrath, Yue Pan, Joel Pinto, Luca Rubini, Miguel Ruiz, Gagandeep Singh, Fabian Stemmer, Weiyi Sun, Paul Vozila, Thomas Lin, and Ranjani Ramamurthy. 2020a. Generating Medical Reports from Patient-Doctor Conversations Using Sequence-to-Sequence Models. In *Proceedings of the First Workshop on Natural Language Processing for Medical Conversations*, pages 22–30, Online. Association for Computational Linguistics.

Seppo Enarvi, Marilisa Amoia, Miguel Del-Agua Teba, Brian Delaney, Frank Diehl, Stefan Hahn, Kristina Harris, Liam McGrath, Yue Pan, Joel Pinto, et al. 2020b. Generating medical reports from patient-doctor conversations using sequence-to-sequence models. In *Proceedings of the first workshop on natural language processing for medical conversations*, pages 22–30.

Gregory Finley, Erik Edwards, Amanda Robinson, Michael Brenndoerfer, Najmeh Sadoughi, James Fone, Nico Axtmann, Mark Miller, and David Suendermann-Oeft. 2018. An automated medical scribe for documenting clinical encounters. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 11–15, New Orleans, Louisiana. Association for Computational Linguistics.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. SWITCHBOARD: telephone speech corpus for research and development. In *Proceedings of the 1992 IEEE international conference on Acoustics, speech and signal processing - Volume 1*, ICASSP'92, pages 517–520, USA. IEEE Computer Society.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented transformer for speech recognition.

Xuehai He, Shu Chen, Zeqian Ju, Xiangyu Dong, Hongchao Fang, Sicheng Wang, Yue Yang, Jiaqi Zeng, Ruisi Zhang, Ruoyu Zhang, Meng Zhou, Penghui Zhu, and Pengtao Xie. 2020. MedDialog: Two Large-scale Medical Dialogue Datasets. *arXiv:2004.03329 [cs, stat]*. ArXiv: 2004.03329.

Tobias Hodgson and Enrico Coiera. 2016. Risks and benefits of speech recognition for clinical documentation: a systematic review. *Journal of the American Medical Informatics Association : JAMIA*, 23(e1):e169–e179.

Maree Johnson, Samuel Lapkin, Vanessa Long, Paula Sanchez, Hanna Suominen, Jim Basilakis, and Linda Dawson. 2014. A systematic review of speech recognition technology in health care. *BMC Medical Informatics and Decision Making*, 14:94.

Anirudh Joshi, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2020. Dr. summarize: Global summarization of medical dialogue by exploiting local structures. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 3755–3763.

Zeqian Ju, Subrato Chakravorty, Xuehai He, Shu Chen, Xingyi Yang, and Pengtao Xie. 2020. Coviddialog: Medical dialogue datasets about covid-19. *https://github.com/UCSD-AI4H/COVID-Dialogue*.

Nazmul Kazi, Matt Kuntz, Upulee Kanewala, and Indika Kahanda. 2020. Dataset for automated medical transcription.

Suyoun Kim. 2020. *End-to-End Speech Recognition on Conversations*. thesis, Carnegie Mellon University.

Jodi Kodish-Wachs, Emin Agassi, Patrick Kenny, and J. Marc Overhage. 2018. A systematic comparison of contemporary automatic speech recognition engines for conversational clinical speech. *AMIA Annual Symposium Proceedings*, 2018:683–689.

Samuel Kriman, Stanislav Beliaev, Boris Ginsburg, Jocelyn Huang, Oleksii Kuchaiev, Vitaly Lavrukhin, Ryan Leary, Jason Li, and Yang Zhang. 2020. Quartznet: Deep automatic speech recognition with 1d time-channel separable convolutions. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6124–6128. IEEE.

Kundan Krishna, Sopan Khosla, Jeffrey Bigham, and Zachary C. Lipton. 2021. Generating SOAP notes from doctor-patient conversations using modular summarization techniques. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint*

Conference on Natural Language Processing (Volume 1: Long Papers), pages 4958–4972, Online. Association for Computational Linguistics.

Yaa A. Kumah-Crystal, Claude J. Pirtle, Harrison M. Whyte, Edward S. Goode, Shilo H. Anders, and Christoph U. Lehmann. 2018. Electronic Health Record Interactions through Voice: A Review. *Applied Clinical Informatics*, 09(3):541–552. Publisher: Georg Thieme Verlag KG.

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

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Zhengyuan Liu, Angela Ng, Sheldon Lee, Ai Ti Aw, and Nancy F Chen. 2019. Topic-aware pointer-generator networks for summarizing spoken conversations. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 814–821. IEEE.

Sean MacAvaney, Sajad Sotudeh, Arman Cohan, Nazli Goharian, Ish Talati, and Ross W Filice. 2019. Ontology-aware clinical abstractive summarization. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1013–1016.

Anirudh Mani, Shruti Palaskar, and Sandeep Konam. 2020. Towards Understanding ASR Error Correction for Medical Conversations. In *Proceedings of the First Workshop on Natural Language Processing for Medical Conversations*, pages 7–11, Online. Association for Computational Linguistics.

Derek Miller. 2019. Leveraging bert for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165*.

Sabine Molenaar, Lientje Maas, Verónica Burriel, Fabiano Dalpiaz, and Sjaak Brinkkemper. 2020. Medical Dialogue Summarization for Automated Reporting in Healthcare. *Advanced Information Systems Engineering Workshops*, 382:76–88.

Francesco Moramarco, Alex Papadopoulos Korfiatis, Aleksandar Savkov, and Ehud Reiter. 2021. A preliminary study on evaluating consultation notes with post-editing. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 62–68.

Francesco Moramarco, Alex Papadopoulos Korfiatis, Mark Perera, Damir Juric, Jack Flann, Ehud Reiter, Anya Belz, and Aleksandar Savkov. 2022. (in press): Human evaluation and correlation with automatic metrics in consultation note generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

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

Patricia F Pearce, Laurie Anne Ferguson, Gwen S George, and Cynthia A Langford. 2016. The essential soap note in an ehr age. *The Nurse Practitioner*, 41(2):29–36.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, CONF. IEEE Signal Processing Society.

Juan C Quiroz, Liliana Laranjo, Ahmet Baki Kocaballi, Agustina Briatore, Shlomo Berkovsky, Dana Rezazadegan, and Enrico Coiera. 2020. Identifying relevant information in medical conversations to summarize a clinician-patient encounter. *Health Informatics Journal*, 26(4):2906–2914. Publisher: SAGE Publications Ltd.

Sai P. Selvaraj and Sandeep Konam. 2021. Medication Regimen Extraction from Medical Conversations. In Arash Shaban-Nejad, Martin Michalowski, and David L. Buckeridge, editors, *Explainable AI in Healthcare and Medicine: Building a Culture of Transparency and Accountability*, Studies in Computational Intelligence, pages 195–209. Springer International Publishing, Cham.

Hagen Soltau, Mingqiu Wang, Izhak Shafran, and Laurent El Shafey. 2021. Understanding Medical Conversations: Rich Transcription, Confidence Scores & Information Extraction. *arXiv:2104.02219 [cs]*. ArXiv: 2104.02219.

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

Wen-wai Yim and Meliha Yetisgen-Yildiz. 2021. Towards automating medical scribing: Clinic visit dialogue2note sentence alignment and snippet summarization. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 10–20.

Longxiang Zhang, Renato Negrinho, Arindam Ghosh, Vasudevan Jagannathan, Hamid Reza Hassanzadeh, Thomas Schaaf, and Matthew R Gormley. 2021.

Leveraging pretrained models for automatic summarization of doctor-patient conversations. *arXiv preprint arXiv:2109.12174*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yuhao Zhang, Derek Merck, Emily Tsai, Christopher D Manning, and Curtis Langlotz. 2020. Optimizing the factual correctness of a summary: A study of summarizing radiology reports. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5108–5120.

Appendix

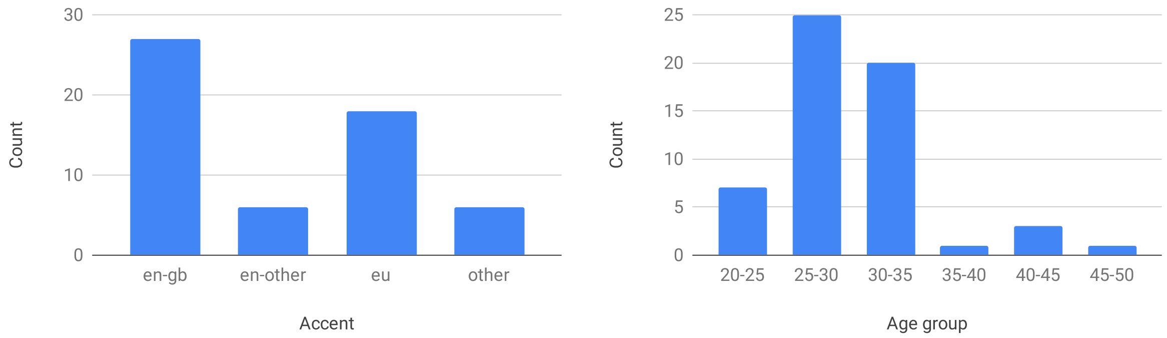


Figure A.1: Accent and age group distributions for patients in the 57 mock consultations.

Demographics (age, gender):
23 year old female
Presenting Complaint: Lower abdominal pain Duration of symptoms: 2 days
History, on open questioning: Have a terrible ache in my lower tummy and feeling hot and sweaty.
Symptoms and risk factors: There is some blood in the urine – pink colour Pain below belly button Feeling nauseated but no vomiting Going to the toilet a little more often but drinking lots of fluids No urine urgency or pain when passing urine. Was constipated until 1 week ago but that has cleared up now Had sexual intercourse 4 days ago No new sexual partner since last STI screen 6 months ago No vaginal discharge Has Implanon contraceptive implant for 1 year No change in vaginal bleeding No loin pain Activities of daily living: No problems performing daily activities Family history: nil Past Medical History: nil Drug History: Implanon Allergies: Amoxicillin

Table A.1: Example clinical case card for a Urinary Tract Infection. Mock patients were given a case card and asked to study it before consulting with the clinician.

Human Transcription	**Google Speech-to-text**
Doctor: Hello? **Patient:** Hello. Can you hear me well? **Doctor:** Uh uh yes. I think. It's a bit better. It's a bit, it's a bit, it's not very clear. But let's continue anyway. **Patient:** OK. **Doctor:** Uh, OK. Let's start again. So how can I help you sir? **Patient:** Yes. So, it's been a few days now. I have like a sore, and a red skin. It's kind of, it's really itchy, and it's like super annoying. So I'd like to find something quick to solve it. **Doctor:** OK. No, no problem. I'm happy to help. Um whereabouts in your skin is it affected? **Patient:** Uh, mostly like my chest, my, my hands, my arms. Like, like really, it's it's super annoying. Like it's itching a lot, like all the time. And I can't even sleep at night. I really need something quickly to, to solve it. Because even at work I, I can, when I'm in a meeting and I have to, like uh think about my work, I can't focus, I can't actually focus on my work. It's really annoying because I can't actually think about, uh, what I have to say. I'm always like, uh, disturbed by this disease.	**Doctor:** Hello. **Patient:** Hello, can you hear me wet? **Doctor:** Yes, I think it's a bit better. It's a bit. It's a bit. It's not very clear. But let's continue. Anyway, **Patient:** Okay. **Doctor:** okay, let's talk again. So, how can I help you, sir? **Patient:** Yes, so it's been a few days now. I have like a sore and the Redskin it's kind of it's really itchy and it's like super annoying. **Doctor:** Okay. **Patient:** So I'd like to find something quick to serve it. **Doctor:** No, no problem. Happy to help whereabouts of your skin is affected. **Patient:** Mostly like my chest my my hands my arms like agree. It's super annoying like it's itching a lot like all the time and I can't even sleep at night. Like I really need something quickly to study because even at work I like when I'm in the meeting and I have to like think about my work Focus like actually focus on my work. It's **Doctor:** Yeah. **Patient:** really annoying because I can actually think about what happened say, I'm always like disturbed by this disease.
* * *	* * *
Doctor: OK. OK. So it's something for you to think about. you can get different types of antihistamines. I can give you something a little bit stronger today as well. Um, something like Fexofenadine, which I can give to you today. It's definitely worth trying, and it's not going to do you any harm. **Patient:** OK. **Doctor:** Um but I think using the steroids and the emollients, um on a regular basis Uh over the next week to ten days, should hopefully control your symptoms. But do come back and see me next week, if things don't get better. **Patient:** That sounds good. **Doctor:** OK? Um do you have any questions for me? **Patient:** Uh, no that's it. Thank you very much. Bye. Thank you as well. Bye.	**Doctor:** It didn't okay. So something for you to think about a you can get different types of and system means I can give you something Little Bit Stronger today as well **Patient:** Okay. **Doctor:** something like fix the penalty in which I can give to you today. It's definitely worth trying it's not gonna do you any harm but I say anything using the steroids and the emollients on a regular basis over the next week to 10 days should hopefully care control your symptoms, but do come back and see me next week if things don't get better. **Patient:** That sounds good. **Doctor:** Okay any questions for me? **Patient:** And now that's it. **Doctor:** Okay. Well, I wish you all the best. **Patient:** Thank you very much. **Doctor:** Hope you have a good day. **Patient:** Bye-bye.

Table A.2: An example of a human transcript and a Google Speech-to-text transcript for one of the mock consultations. Both are partial.

Human Note	Hx: 1 week history of spontaneous elbow swelling left. Not painful. No trauma. No FH of rheumatological disease- NB pt says he has been old he has OA previously by doctors- ? need to confirm this Works in a desk job Not happened before Otherwise well- PMHx: nil of note FH: nil of note DH: not on any medication, allergic to peanuts SH: exercises regularly, active Ex: looks well, not in pain. Mild erythema and minimal swelling (if any) around olecranon process left elbow Imp: possible bursitis Plan: for NSAIDs- usual advice re SE For rheum bloods: esr, crp, fbc, rheum factor and urate Review thereafter in person/ via video To contact us back in interim if any deterioration/concerns- pt warned re symptoms of septic arthritis.
BART-CNN	Doctor Deen Mirza from GP at Hand sees John Smith. John says he has a weird swelling on his left elbow. He also says he is allergic to peanuts. Deen takes a look at John's elbow to see if there is anything wrong with it. Do you have any other illnesses at all?
BERT-ext	Before we start your appointment, could you please tell me your first name and your date of birth. And I was born on the fifth of April, , nineteen seventy three. But it's just, just a bit, a bit weird, to see that. , and , , in terms of your job, do you do anything physical? so you know you said you think you've got , , osteoarthritis. and, do you have any other illnesses at all? , I run regularly, like two, three times a week. , what I think we should do is, I think you should be on some anti-inflammatory medication, in the, in the first instance. And, there'll be instructions within that pack, about where to go to get those blood tests done. and , your, your joint doesn't look like that. However, if your, the elbow was to become very red, very painful, , and the redness was to spread or become , you know more intense. That would require more immediate assessment, more immediate treatment. do you, do you think it's something dangerous? Like something, like could I die from that, or is it, is it No. that's four hundred milligrams, two times a day. Maybe within a , actually you know, the follow-up appointment doesn't have to be face-to-face, if it's more convenient for you do, to do it over the phone, we can do that over the phone, , over video. We can do that as well, that's, that's your call.
Random	Sure. No, no I haven't noticed that before. OK, OK, great. Yes, a few years ago. do you, do you think it's something dangerous? Fantastic. But you contact us, , after you've had the blood test done, and we can review things then, OK. OK. OK, yeah that sounds good. OK. -. , yeah, no, I'm, think I'm healthy. . So, , this, this is not the case right now. I run regularly, like two, three times a week. don't need to worry. All right then, OK. , take care then.
BART-finet	You have a problem with your left elbow. 1 week ago noticed a weird swelling on the left elbow. Not painful at all, but slightly warm, slightly warm. No pain, no swelling, no fluid in the elbow. No injury. No previous history of this. No injury to the elbow. NKDA. SH: Mobile and active, exercise 2-3 times a week, running. Osteoarthritis of the elbow. You should start the treatment you have been prescribed. You should begin the treatment prescribed as we discussed. You may want to take some ibuprofen or paracetamol in addition to any prescribed medication.

Table A.3: Examples of a human written note and automatically generated notes with the four baseline models.

UniGDD: A Unified Generative Framework for Goal-Oriented Document-Grounded Dialogue [*]

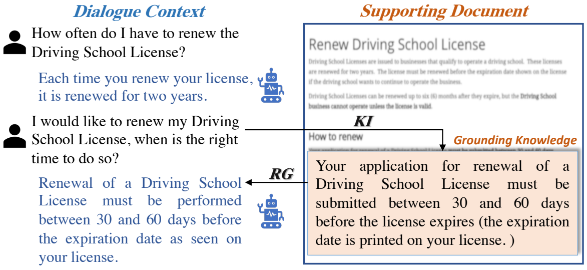
Chang Gao, Wenxuan Zhang, and Wai Lam
The Chinese University of Hong Kong
{gaochang,wxzhang,wlam}@se.cuhk.edu.hk

Abstract

The goal-oriented document-grounded dialogue aims at responding to the user query based on the dialogue context and supporting document. Existing studies tackle this problem by decomposing it into two sub-tasks: knowledge identification and response generation. However, such pipeline methods would unavoidably suffer from the error propagation issue. This paper proposes to unify these two sub-tasks via sequentially generating the grounding knowledge and the response. We further develop a prompt-connected multi-task learning strategy to model the characteristics and connections of different tasks and introduce linear temperature scheduling to reduce the negative effect of irrelevant document information. Experimental results demonstrate the effectiveness of our framework.

1 Introduction

Recent years have seen significant progress in goal-oriented dialogues (Loshchilov and Hutter, 2017; Wen et al., 2017; Wu et al., 2019; Hosseini-Asl et al., 2020; Peng et al., 2021), which aim at assisting end users in accomplishing certain goals via natural language interactions. However, due to the lack of external knowledge, most goal-oriented dialogue systems are restricted to providing information that can only be handled by given databases or APIs (Kim et al., 2020) and completing certain tasks in a specific domain such as restaurant booking. To address this challenge, goal-oriented document-grounded dialogue has been proposed to leverage external documents as the knowledge source to assist the dialogue system in satisfying users' diverse information needs (Feng et al., 2020; Wu et al., 2021).



How often do I have to renew the Driving School License?

Each time you renew your license, it is renewed for two years.

I would like to renew my Driving School License, when is the right time to do so?

Renewal of a Driving School License must be performed between 30 and 60 days before the expiration date as seen on your license.

Renew Driving School License

Driving School Licenses are issued to businesses that qualify to operate a driving school. These licenses are renewed for two years. The license must be renewed before the expiration date shown on the license if the driving school wants to continue to operate the business.

Driving School Licenses can be renewed up to six (6) months after they expire, but the **Driving School business cannot operate unless the license is valid.**

How to renew

Your application for renewal of a Driving School License must be submitted between 30 and 60 days before the license expires (the expiration date is printed on your license.)

Figure 1: An example of the goal-oriented document-grounded dialogue problem.

As shown in Figure 1, the goal-oriented document-grounded dialogue problem is commonly formulated as a sequential process including two sub-tasks: knowledge identification (KI) and response generation (RG) (Feng, 2021). Given the dialogue context and supporting document, knowledge identification aims to identify a text span in the document as the grounding knowledge for the next agent response, which is often formulated as a conversational reading comprehension task (Feng, 2021; Wu et al., 2021). Response generation then aims at generating a proper agent response according to the dialogue context and the selected knowledge. Therefore, one straightforward solution for this problem is to use two models to conduct KI and RG in a pipeline manner (Daheim et al., 2021; Kim et al., 2021; Xu et al., 2021; Chen et al., 2021; Li et al., 2021). However, such pipeline methods fail to capture the interdependence between KI and RG. As a result, error propagation is a serious problem. The problem is more pronounced in low-resource scenarios, where accurate knowledge identification is difficult due to limited data, making it harder to generate appropriate responses.

To address the aforementioned issue, we propose a **Uni**fied generative framework for **G**oal-oriented **D**ocument-grounded **D**ialogue (**UniGDD**). Given the dialogue context and associated document, instead of treating KI and RG as two separate processes, we tackle them simultaneously via sequen-

[*] The work described in this paper is substantially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Code: 14200620).

tially generating the grounding knowledge and the agent response. Therefore, the inherent dependencies between these two sub-tasks can be naturally modeled. On one hand, the generation of the agent response depends not only on the dialogue context and external document but also on the identified knowledge, forcing the model to focus on the specific knowledge. On the other hand, the generation of the grounding knowledge receives the supervision signal from the agent response when training, leading to more accurate knowledge identification.

Although KI and RG can be unified with the proposed generative method, they have different characteristics. Generating the grounding knowledge is similar to copying appropriate sentences from the document, while generating the response needs more effort to make the response coherent with the dialogue and consistent with the grounding knowledge. Therefore, in addition to the main task that uses the concatenation of the grounding knowledge and response as the target sequence, we introduce the generation of the grounding knowledge and the generation of the response as two auxiliary tasks in the same framework to force the model to capture their characteristics so as to perform well on them as well. Moreover, inspired by the recent success in prompt learning for pre-trained models (Li and Liang, 2021; Lester et al., 2021; Liu et al., 2021), we design prompts for these three tasks to guide the model on what to generate for each task. These prompts can naturally connect these tasks via indicating the model that each auxiliary task aims to generate a part of the target sequence of the main task. Through this prompt-connected multi-task learning strategy, the model can capture the characteristics of different tasks as well as exploit the connections between them.

In addition, for a particular user query in the goal-oriented dialogue, the selected knowledge and generated response need to be specific, while the generation conditions on a relatively long document. Thus, much information in the input document is irrelevant. To tackle this problem, we introduce linear temperature scheduling to make the attention distribution to the input document gradually sharper during the training process in order to enable the model to learn to pay more attention to the relevant content.

Our contributions are summarized as follows: (1) We propose a unified generative framework for the goal-oriented document-grounded dialogue. (2) We develop a prompt-connected multi-task learning strategy to exploit the characteristics and connections of different tasks and introduce linear temperature scheduling to enable the model to pay more attention to relevant information. (3) Our framework advances state-of-the-art methods on the concerned task, especially in low-resource scenarios.

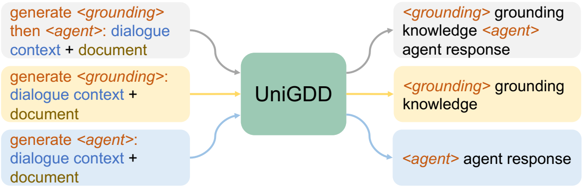

generate *<grounding>* then *<agent>*: dialogue context + document
generate *<grounding>*: dialogue context + document
generate *<agent>*: dialogue context + document
UniGDD
<grounding> grounding knowledge *<agent>* agent response
<grounding> grounding knowledge
<agent> agent response

Figure 2: Overview of our framework.

2 Our UniGDD framework

UniGDD is a multi-task generative framework for the goal-oriented document-grounded dialogue problem.

Main Task Given the dialogue context $C = (u_1, a_1, \ldots, u_{t-1}, a_{t-1}, u_t)$ and grounding document D, where u_i is the i-th user utterance and a_i is the i-th agent utterance, our main task aims to generate the target sequence $Y = (k_t, a_t)$, where k_t is the grounding knowledge from D and a_t is the response to u_t. Specifically, for the example in Figure 1, the input and output of the main task are as follows:

> Input: generate *<grounding>* then *<agent>*: *<user>* I would like to renew ... ? *<agent>* Each time you ... *<user>* How often do ... ? *<title>* Renew Driving School License *</title>* ... Your application for renewal ...
> Output: *<grounding>* Your application for ... *<agent>* Renewal of a Driving ...

We use different special tokens to identify different elements in the input and output. For example, we add *"<user>"* in front of each user utterance, *"<agent>"* in front of each agent utterance, and *"<grounding>"* in front of the grounding knowledge. The prompt "generate *<grounding>* then *<agent>*:" is added to the dialogue context and supporting document to form the input and guide the model to generate the grounding knowledge and the response in order. The input-to-target generation can be modeled with a pre-trained encoder-decoder model $\mathcal{M} : (C, D, TP) \rightarrow (k_t, a_t)$ such as T5 (Raffel et al., 2020), where TP is the task prompt.

Prompt-Connected Multi-Task Learning We introduce two auxiliary tasks to steer our framework to model the respective characteristics of knowledge identification and response generation. Given the dialogue context C and grounding document D, these two tasks aim to generate the grounding knowledge k_t and the response a_t with the same model $\mathcal{M}$. As depicted in Figure 2, we construct prompts "generate <*grounding*>:" and "generate <*agent*>:" for them. These prompts indicate the model that the goals of the two auxiliary tasks are to generate the first part and the second part of the target sequence of the main task, respectively. As a result, the connections between different tasks are naturally modeled. Instead of using discrete language phrases, we randomly initialize the embeddings of those special tokens in the prompts and train them end-to-end to better encode the characteristics and connections of these tasks.

Linear Temperature Scheduling For a specific user query in the dialogue, many document contents are actually irrelevant. To force the model to pay less attention to the irrelevant parts, we propose a linear temperature scheduling strategy to make the attention distribution of cross-attention gradually sharper during the training process. Specifically, we design the softmax function in the cross-attention module of each decoder layer as follows:

$$a_i = \frac{\exp\left(z_i/\tau\right)}{\sum_j \exp\left(z_j/\tau\right)} \tag{1}$$

$$\tau = (\tau_e - \tau_s)\frac{S_c}{S_{total}} + \tau_s \tag{2}$$

where a_i is the attention weight for the i-th input token, z_i is the logit for the i-th input token, S_c is the current training step, S_{total} is the total training steps, τ_s and τ_e are the starting and ending temperature respectively, $\tau_e < \tau_s$, and $0 < \tau_e < 1$. Compared with the original cross-attention module, the ending temperature $0 < \tau_e < 1$ leads to a sharper attention distribution, giving more attention weight to the relevant content.

Training The model is trained with a maximum likelihood objective. Given the training example $e = (C, D, TP, Y)$, the objective L_θ is defined as

$$\mathcal{L}_\theta = -\sum_{i=1}^{n} \log P_\theta\left(Y_i \mid Y_{<i}, C, D, TP\right) \tag{3}$$

where θ is the model parameters, TP is the task prompt, Y is the target sequence, and n is the length of Y. We mix the data of the main task and two auxiliary tasks for training.

Models	EM	F1
BERTQA	42.2	58.1
BERT-PR-large	56.3	70.8
RoBERTa-PR-large	65.6	77.3
Multi-Sentence	59.5	68.8
DIALKI ($\mathcal{L}_{next}$ only)	60.4	71.2
DIALKI	65.9	74.8
UniGDD-base	65.6	76.8
UniGDD-large	**66.9**	**77.5**

Table 1: Results on knowledge identification.

Models	BLEU
DIALKI+BART-base	25.8
RoBERTa-PR-large+BART-base	39.6
RoBERTa-large+T5-base	40.7
UniGDD-base	42.8
UniGDD-large	**42.9**

Table 2: Results on response generation.

Inference After training, for each pair of dialogue context and document (C, D), we generate the target sequence of the main task for obtaining the grounding knowledge k_t and the response a_t.

3 Experiments

3.1 Experimental Setup

Dataset We conduct experiments on the goal-oriented document-grounded dialogue dataset Doc2Dial (Feng, 2021), which is adopted by the DialDoc21 shared task[1]. It contains 3,474 dialogues with 44,149 turns for training and 661 dialogues with 8539 turns for evaluation[2].

Evaluation Metrics Following Feng (2021), we use Exact Match (EM) and token-level F1 for knowledge identification and BLEU (Papineni et al., 2002; Post, 2018) for response generation.

Baselines For knowledge identification, we compare UniGDD with several strong baselines, including BERTQA (Devlin et al., 2019), BERT-PR (Daheim et al., 2021), RoBERTa-PR (Daheim et al., 2021), Multi-Sentence (Wu et al., 2021), and DIALKI (Wu et al., 2021). These models formulate knowledge identification as the machine reading comprehension task and extract the grounding span

[1] https://github.com/doc2dial/sharedtask-dialdoc2021

[2] Since we cannot access the test set, we report results on the development set for comparison.

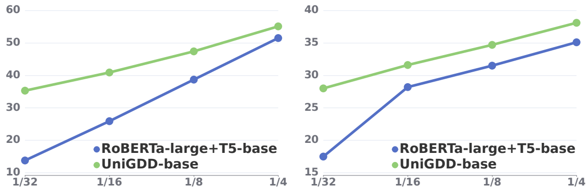
from the document. For response generation, we compare UniGDD with several pipeline methods, including DIALKI+BART (Wu et al., 2021) that uses DIALKI to conduct knowledge identification, followed by BART (Lewis et al., 2020) to conduct response generation and RoBERTa-PR+BART (Daheim et al., 2021). We also build a strong baseline model RoBERTa+T5 which uses the same pretrained generative model as ours.

Implementation Details We report results of UniGDD with two model sizes: UniGDD-base and UniGDD-large, which are initialized with pretrained T5-base and T5-large models (Raffel et al., 2020), respectively. We adopt the implementation from Hugging Face Transformers (Wolf et al., 2020). We set the max input length to 2560. Any sequence over 2560 tokens will be truncated. For training, we use the AdamW (Loshchilov and Hutter, 2019) optimizer with an initial learning rate of 10^{-4} and a linear learning rate decay scheduler. We train 10 epochs for single-task learning and 5 epochs for multi-task learning. For decoding, we use beam search, and the beam size is 2. For linear temperature scheduling, we set the starting temperature $\tau_s = 1$ and choose the best ending temperature from $\{0.5, 0.6, 0.7, 0.8, 0.9\}$. For our constructed baseline RoBERTa+T5 for response generation, we use RoBERTa-large and T5-base and adopt the implementation from the DialDoc21 shared task.

3.2 Results

The results on knowledge identification and response generation are shown in Table 1 and Table 2, respectively. Our UniGDD framework outperforms all the baselines on two sub-tasks. On the knowledge identification task, UniGDD-base can obtain comparable results to previous state-of-the-art methods. With a larger model size, UniGDD-large achieves new state-of-the-art performance. On the response generation task, UniGDD obtains a marked improvement over all pipeline methods. This verifies our assumption that our unified generative framework can alleviate the error propagation problem of pipeline approaches.

Effect of Prompt-Connected Multi-task Learning (PCMTL) and Linear Temperature Scheduling (LTS) To verify the effectiveness of PCMTL and LTS, we first remove PCMTL (i.e., training with the main task only), and the performance of UniGDD-base on two tasks decreases to 65.2 EM, 76.3 F1, and 42.3 BLEU, showing that PCMTL endows the model with the ability of modeling the characteristics and connections of different tasks and achieving better generation. Further removing LTS, the performance drops to 64.7 EM, 76.0 F1, and 41.7 BLEU. This indicates that LTS can guide the model to pay more attention to relevant content during generation and bring improvements on two sub-tasks.



(a) EM (b) BLEU

Figure 3: Experimental results on knowledge identification and response generation in low-resource scenarios

Effect of Connected Prompts (CP) To examine whether CP can capture the connections of different tasks, we use an alternative approach that employs task-independent prompts "<Task1>:", "<Task2>:", and "<Task3>:" to specify each task for comparison. As in the case of CP, we randomly initialize the embeddings of these three special tokens. With these prompts, UniGDD-base obtains 64.9 EM, 76.2 F1, and 42.3 BLEU, which performs worse than using CP. This indicates that CP enables the model to take advantage of the connections between the three tasks.

Low-Resource Setting To evaluate the model in low-resource scenarios, we randomly shuffle the training set and then take 1/32, 1/16, 1/8, and 1/4 of the data for training. Figure 3 shows the results of UniGDD-base and the best-performing pipeline baseline RoBERTa-large+T5-base on the four low-resource training splits. Generally, our framework performs substantially better than the pipeline method on both tasks. Particularly, when there is only 1/32 training data, UniGDD-base obtains more than 20 and 10 absolute points improvement over the pipeline approach on EM and BLEU, respectively.

Case Study Figure 4 shows a real case including the dialogue context, supporting document, and the responses generated by the pipeline method and our proposed UniGDD framework. It can be observed that our framework identifies accurate knowledge from the supporting document and thus provides a

Dialogue Context

👤 I filled out all of the information in the Retirement Estimator and it took a long time. When I came back from answering the door, all of the information was gone. What happened?

🤖 Oh that's too bad. Were you gone for a long time?

👤 Yes I guess I was.

Supporting Document

…… How Long Can You Stay On Each Page? *For security reasons, there are time limits for viewing each page. You will receive a warning after 25 minutes without doing anything, and you will be able to extend your time on the page*. After the third warning on a page, you must move to another page. If you do not, your time will run out and your work on that page will be lost.

Response

RoBERTa-large+T5-base

🤖 Do you have any more questions about the Retirement Estimator?

UniGDD-base

🤖 For security reasons, there are time limits for viewing each page. You will receive a warning after 25 minutes without doing anything and you will be able to extend your time on the page.

Ground Truth

🤖 For reasons of security, there are time limits for viewing each page.

Figure 4: A case from the development set.

proper and informative response about the reasons for the problem the user encounters. In contrast, the pipeline method only gives a relatively general response that is not suitable in this case.

3.3 Human Evaluation

We randomly sample 100 evaluation instances. For each instance, given the dialogue context and grounding document, three human annotators are asked to conduct a pairwise comparison between the response generated by UniGDD-base and the one generated by the pipeline baseline RoBERTa-large+T5-base in terms of two aspects: (1) *Relevance*: which response is more relevant and appropriate to the user query? (2) *Informativeness*: which response is more informative? Results are shown in Table 3. Compared with the pipeline method, our framework can reduce error propagation, resulting in more relevant and appropriate responses. Moreover, our framework has a clear advantage over the baseline in terms of Informativeness since it can utilize rich document context during the generation.

	Win	Tie	Lose
Relevance	26	64	10
Informativeness	23	69	8

Table 3: UniGDD-base vs RoBERTa-large+T5-base. The numbers indicate how many instances there are in each case.

4 Conclusion

Our UniGDD framework unifies knowledge identification and response generation and models their characteristics via a multi-task generative modeling strategy. Both automatic evaluation and human evaluation demonstrate the effectiveness of our framework.

References

Xi Chen, Faner Lin, Yeju Zhou, Kaixin Ma, Jonathan Francis, Eric Nyberg, and Alessandro Oltramari. 2021. Building goal-oriented document-grounded dialogue systems. In *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*, pages 109–112, Online. Association for Computational Linguistics.

Nico Daheim, David Thulke, Christian Dugast, and Hermann Ney. 2021. Cascaded span extraction and response generation for document-grounded dialog. In *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*, pages 57–62, Online. Association for Computational Linguistics.

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

Song Feng. 2021. DialDoc 2021 shared task: Goal-oriented document-grounded dialogue modeling. In *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*, pages 1–7, Online. Association for Computational Linguistics.

Song Feng, Hui Wan, Chulaka Gunasekara, Siva Patel, Sachindra Joshi, and Luis Lastras. 2020. doc2dial: A goal-oriented document-grounded dialogue dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8118–8128, Online. Association for Computational Linguistics.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *Advances in Neural Information Processing Systems*, volume 33, pages 20179–20191. Curran Associates, Inc.

Boeun Kim, Dohaeng Lee, Sihyung Kim, Yejin Lee, Jin-Xia Huang, Oh-Woog Kwon, and Harksoo Kim. 2021. Document-grounded goal-oriented dialogue systems on pre-trained language model with diverse input representation. In *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*, pages 98–102, Online. Association for Computational Linguistics.

Seokhwan Kim, Mihail Eric, Karthik Gopalakrishnan, Behnam Hedayatnia, Yang Liu, and Dilek Hakkani-Tur. 2020. Beyond domain APIs: Task-oriented conversational modeling with unstructured knowledge access. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 278–289, 1st virtual meeting. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Jiapeng Li, Mingda Li, Longxuan Ma, Wei-Nan Zhang, and Ting Liu. 2021. Technical report on shared task in DialDoc21. In *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*, pages 52–56, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Ilya Loshchilov and Frank Hutter. 2017. Learning end-to-end goal-oriented dialog. In *International Conference on Learning Representations*.

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

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

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. Soloist: Building Task Bots at Scale with Transfer Learning and Machine Teaching. *Transactions of the Association for Computational Linguistics*, 9:807–824.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

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

Tsung-Hsien Wen, Yishu Miao, Phil Blunsom, and Steve Young. 2017. Latent intention dialogue models. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3732–3741. PMLR.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819, Florence, Italy. Association for Computational Linguistics.

Zeqiu Wu, Bo-Ru Lu, Hannaneh Hajishirzi, and Mari Ostendorf. 2021. DIALKI: Knowledge identification in conversational systems through dialogue-document contextualization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1852–1863, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yan Xu, Etsuko Ishii, Genta Indra Winata, Zhaojiang Lin, Andrea Madotto, Zihan Liu, Peng Xu, and Pascale Fung. 2021. CAiRE in DialDoc21: Data augmentation for information seeking dialogue system. In *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*, pages 46–51, Online. Association for Computational Linguistics.

DMIX: Adaptive Distance-aware Interpolative Mixup

Ramit Sawhney[†][*], **Megh Thakkar**[§][*], **Shrey Pandit**[§][*], **Ritesh Soun**[♣]
Di Jin[★], **Diyi Yang**[△], **Lucie Flek**[†]
[†]Conversational AI and Social Analytics (CAISA) Lab, University of Marburg
[§]BITS, Pilani
[♣]Sri Venkateswara College, DU
[★]Amazon Alexa AI
[△]Georgia Institute of Technology
rsawhney@mathematik.uni-marburg.de, lucie.flek@uni-marburg.de

Abstract

Interpolation-based regularisation methods such as Mixup, which generate virtual training samples, have proven to be effective for various tasks and modalities. We extend Mixup and propose DMIX, an adaptive distance-aware interpolative Mixup that selects samples based on their diversity in the embedding space. DMIX leverages the hyperbolic space as a similarity measure among input samples for a richer encoded representation. DMIX achieves state-of-the-art results on sentence classification over existing data augmentation methods on 8 benchmark datasets across English, Arabic, Turkish, and Hindi languages while achieving benchmark F1 scores in 3 times less number of iterations. We probe the effectiveness of DMIX in conjunction with various similarity measures and qualitatively analyze the different components. DMIX being generalizable, can be applied to various tasks, models and modalities.

1 Introduction

Deep learning models, though effective for many applications are prone to overfitting in absence of sufficient training data. Data augmentation techniques can efficiently use this limited training data (Liu et al., 2021; Shi et al., 2020). Interpolation-based augmentation techniques such as Mixup (Zhang et al., 2018) have shown improved performance across different modalities. Mixup over latent representations of inputs leads to further improvements (Chen et al., 2020a). However, Mixup does not account for the spatial distribution of dataset samples, but choosing samples randomly for interpolation-based augmentation.

While randomization in Mixup helps, augmenting Mixup's sample selection strategy with logic based on the similarity of the samples to be mixed can lead to improved generalization (Chen et al., 2020b). The relative spatial position of samples can be leveraged to produce more suitable synthetic inputs for training underlying models (Xu et al., 2021). Further, natural language possesses hierarchical structures and complex geometries, which the standard Euclidean space cannot capture effectively (Ganea et al., 2018). Hyperbolic geometry presents a solution in defining similarity between latent representations (Tifrea et al., 2019).

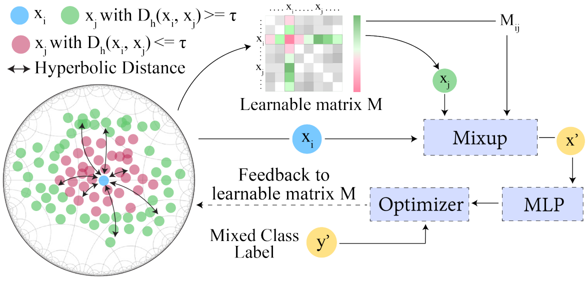


Figure 1: Overview of DMIX showing the sample selection based on the hyperbolic distance and using distance matrix $\mathbf{M}$ to perform interpolation.

We propose DMIX, an adaptive distance-aware interpolative data augmentation method. Instead of choosing random inputs from the complete training distribution as in the case of Mixup, DMIX samples instances based on the (dis)similarity between latent representations of samples in the hyperbolic space. Furthermore, DMIX performs interpolations with trainable pair-wise parameters derived from the spatial distribution of the samples rather than sampling mixing ratios randomly from standard distributions, making it adaptive for pair-wise interpolation. Our contributions are:

- We propose DMIX, a novel adaptive distance-aware interpolative regularization method developed over the spatial distribution of dataset sampled in the hyperbolic space.
- DMIX outperforms existing interpolative data augmentation baselines for 8 benchmark sentence classification tasks across four languages.
- DMIX achieves threshold F1 scores with 3 times less number of iterations than random Mixup

[*] Equal contribution.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 606 - 612
May 22-27, 2022 ©2022 Association for Computational Linguistics

while being generalizable across tasks, datasets, and modalities.

2 Methodology

We present an overview of DMIX in Figure 1. We first introduce interpolative Mixup (§2.1), and then formulate DMIX by leveraging the relative sample distribution in the hyperbolic space (§2.2).

2.1 Interpolative Mixup

Given two data samples $x_i, x_j \in X$ with labels $y_i, y_j \in Y$, and $i, j \in [1, N]$, Mixup (Zhang et al., 2018) uses linear interpolation with mixing ratio r to generate the synthetic sample x' and corresponding mixed label y',

$$\begin{aligned} x' &= \text{Mixup}(x_i, x_j) = r \cdot x_i + (1-r) \cdot x_j \\ y' &= \text{Mixup}(y_i, y_j) = r \cdot y_i + (1-r) \cdot y_j \end{aligned} \tag{1}$$

Interpolative Mixup (Chen et al., 2020a) performs linear interpolation over the latent representations of models. Let $f_\theta(\cdot)$ be a model with parameters θ having K layers, $f_{\theta,n}(\cdot)$ denotes the n-th layer of the model and h_n is the hidden space vector at layer n for $n \in [1, K]$ and h_0 denotes the input vector. To perform interpolative Mixup at a layer $k \sim [1, K]$, we calculate the latent representations separately for the inputs for layers before the k-th layer. For input sample x_i, we let h_n^i denote the hidden state representations at layer n,

$$\begin{aligned} h_n^i &= f_{\theta,n}(h_{n-1}^i), \quad n \in [1, k] \\ h_n^j &= f_{\theta,n}(h_{n-1}^j), \quad n \in [1, k] \end{aligned} \tag{2}$$

We then perform Mixup over individual hidden state representations h_k^i, h_k^j from layer k as,

$$h_k = \text{Mixup}(h_k^i, h_k^j) = r \cdot h_k^i + (1-r) \cdot h_k^j \tag{3}$$

The mixed hidden representation h_k is used as the input for the continuing forward pass,

$$h_n = f_{\theta,n}(h_{n-1}); \quad n \in [k+1, K] \tag{4}$$

2.2 DMIX: Distance-aware Mixup

Though Mixup helps generalize models better, it selects samples completely randomly for interpolation. Augmenting the sample selection strategy with intelligence derived from the spatial distribution of the samples to be mixed can lead to improved generalization. Hence, we formulate distance-aware Mixup, or DMIX. To perform DMIX, we first create a learnable matrix $\mathbf{M}_{N\text{x}N}$, which is used to perform Mixup between pair of samples. We use the hyperbolic distance as our similarity metric to initialize matrix $\mathbf{M}$ as it effectively captures the hierarchical structures and complex geometries that natural language text possesses. The hyperbolic distance $\mathcal{D}_h$ between sentence embeddings $e_i = f_\theta(x_i)$ and $e_j = f_\theta(x_j)$ is,

$$\mathcal{D}_h(e_i, e_j) = 2 \tan^{-1}(\|(-e_j) \oplus e_i\|) \tag{5}$$

Here, $\oplus$ represents the Möbius addition $\oplus$ for a pair of points $x, y \in \mathcal{B}$, defined as,

$$x \oplus y := \frac{(1 + 2\langle x, y\rangle + ||y||^2)x + (1 - ||x||^2)y}{1 + 2\langle x, y\rangle + ||x||^2||y||^2} \tag{6}$$

, $\langle ., . \rangle$, $|| \cdot ||$ are Euclidean inner product and norm.

We initialize $\mathbf{M}$ using hyperbolic distance $\mathcal{D}_h$ and normalize it row wise to scale the values,

$$\mathbf{M}_{ij} = \mathcal{D}_h(e_i, e_j); \quad \mathbf{M}_i = \frac{\mathbf{M}_i}{max(\mathbf{M}_i)} \tag{7}$$

Using learnable matrix $\mathbf{M}$, we change the Mixup formulation (Equation 1) for samples i and j and define DMixup as,

$$\text{DMixup}(x_i, x_j) = (1 - \mathbf{M}_{ij}) * x_i + \mathbf{M}_{ij} * x_j \tag{8}$$

DMIX is defined for one sample as compared to Mixup which is defined for two samples. To perform DMIX over a sample x_i, we create a set S_i of the most diverse samples in the dataset based on a threshold. To create this set, we select samples having $\mathbf{M}_{ij}$ above a threshold τ,

$$S_i = \{x_k | x_k \in X, \mathbf{M}_{ik} \geq \tau\} \tag{9}$$

We use τ to control the diversity of the selected samples. $\tau = \text{T} * max(\mathbf{M}_i)$ at each step of the training, where T is a hyperparameter $\in (0, 1)$. To perform DMIX, we operate DMixup over samples x_i and a random sample $x_j \in S_i$,

$$\text{DMIX}(x_i) = \text{DMixup}(x_i, x_j), \quad x_j \in S_i \tag{10}$$

We replace the Mixup operation in Equation 3 with the DMIX operation in Equation 10 to evaluate DMIX. The final hidden state output h_K is passed through a multi-layer perceptron (MLP) g_ϕ for classification. We optimize the network using KL Divergence loss between the final output $g_\phi(h_K)$ and mixed label $y' = \text{DMixup}(y_i, y_j)$, which also trains matrix $\mathbf{M}$ end-to-end.

3 Experimental Setup

We evaluate DMIX on standard English, GLUE, and multi-lingual datasets in 4 languages (Table 1).

Dataset	Language	Classes	Samples
TRAC (2020)	English	3	5,329
TREC-Coarse (2002)	English	6	5,952
TREC-Fine (2002)	English	47	5,952
CoLA (2018)	English	2	10,657
SST-2 (2013)	English	2	12,693
AHS (2018)	Arabic	2	3,950
TTC (2017)	Turkish	6	3,600
HASOC (2019)	Hindi	2	5,983

Table 1: Datasets, languages, # classes and # samples.

Dataset	f_θ	+WMix	+SMix	+TMix	+DMix
TRAC	72.52	73.52	74.20	75.41	**78.67***
TREC-Coarse	97.08	96.10	96.59	97.52	**97.80***
TREC-Fine	86.86	87.13	87.89	90.16	**91.14***
CoLA	84.91	84.95	85.14	85.30	**95.94***
SST-2	90.32	91.34	91.21	91.66	**92.44***
AHS	66.39	67.10	68.30	70.19	**74.98***
TTC	91.10	90.18	91.15	91.30	**92.16***
HASOC	76.13	77.24	76.30	77.44	**80.27***

Table 2: Performance comparison in terms of F1 score of baseline methods with DMIX (average of 10 runs). * shows significant ($p < 0.01$) improvement over TMix.

Dataset	TMix	Euc-DMIX NT	DMIX NT	Euc-DMIX	DMIX
TRAC	75.41	76.52*	78.16*	77.02*	**78.67**$^{*\diamond}$
TREC-Coarse	97.52	97.55	97.66	97.53	**97.80***
TREC-Fine	90.16	89.70	90.20	89.12	**91.14**$^{*\diamond}$
CoLA	85.30	85.73*	86.81*	86.23*	**95.94**$^{*\diamond}$
SST-2	91.05	91.15	92.31*	91.92*	**92.44**$^{*\diamond}$
AHS	70.19	72.23*	74.65*	72.41*	**74.98**$^{*\diamond}$
TTC	91.30	90.66	91.40	91.50	**92.16**$^{*\diamond}$
HASOC	77.44	78.96*	79.96*	79.38*	**80.27**$^{*\diamond}$

Table 3: Ablation study of DMIX with distance constraints using different similarity techniques (average of 10 runs). Improvements are shown with blue . *, $^\diamond$ show significant ($p < 0.01$) improvement over TMix and DMIX-NT, respectively.

3.1 Training Setup

DMIX is performed over a layer randomly sampled from all the layers of the model. We use a learning rate of 2e-5, batch size of 8 and a weight decay of 0.01 for all the combinations, DMIX, DMix-NT, and Mixup. For the baselines, we sample r from a beta distribution following previous works. All hyperparameters were selected based on validation F1-score. We use BERT for English and mBERT for other languages as the base model f_θ for our experiments, and their [CLS] token representation as the sentence embeddings to calculate the distances (Equation 5). Due to resource constraints, we only use $10,000$ samples of SST-2 for training, but do not change the validation and test split.

3.2 Evaluation

We compare DMIX with word-mixup (WMix) and sentence-mixup (SMix) (Guo et al., 2019), and interpolative Mixup (TMix) (Chen et al., 2020a)[1].

F1 We use F1 score to evaluate the classification performance of DMIX and its variants.

Diversity Following Gontijo-Lopes et al. (2020), we use diversity defined as the number of training steps required to obtain a benchmark F1 score.

4 Results and Analysis

4.1 Performance Comparison and Ablation

We observe that distance-constrained Mixup significantly ($p < 0.01$) outperforms all baselines across the datasets (Table 2) validating that similarity-based sample selection improves model performance, likely owing to enhanced diversity or minimizing sparsification across tasks. Within distance-constrained Mixup, we observe that DMIX, the hyperbolic distance variant outperforms Euclidean distance (Euc-DMIX) measures (Table 3). This suggests that the hyperbolic space is more capable of capturing the complex hierarchical information present in sentence representations, leading to better comparisons and sample selection. We also compare DMIX and its variants with their non-trainable versions (denoted by -NT in Table 3). These methods have matrix $\mathbf{M}$ fixed, and only select samples based on their relative positions in the embedding space. We observe that for all variants, the non-trainable counterparts perform poorer than the trainable counterparts, indicating that $\mathbf{M}$ is able to capture sample-specific information relative to other samples, generating more suitable sample selection and mixing ratio for performing interpolative data augmentation.

4.2 Analyzing Convergence of DMIX

We validate *"Does* DMIX *converge faster than TMix?"*. We observe that across all datasets, DMIX achieves a benchmark F1 score in less number of training iterations compared to TMix (Figure 2). Since DMIX selects samples for Mixup in an adaptive distance-aware manner, it is able to generate more diverse and suitable interpolations leading to faster generalization of the underlying base model. DMIX requires 3 times less number of iterations on an average compared to TMix, or

[1] We provide an extended comparison with other baselines in the Appendix.

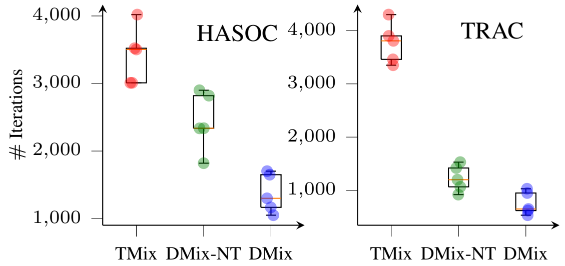


Figure 2: Diversity comparison of TMix with DMix and DMix-NT as number of training steps required to achieve benchmark F1 scores (TRAC:75, HASOC:77).

random Mixup, and hence is more generalizable and effective across languages.

4.3 Impact of Sample Selection and Distance-Aware Mixing Ratio

Model	TTC	TREC-Coarse	AHS
TMix	91.30	97.52	70.19
+ M-Ratio	91.66	96.90	72.43
+ M-Threshold	92.02	97.10	73.31
DMix	**92.16**	**97.80**	**74.98**

Table 4: Ablation study over matrix $\mathbf{M}$ (F1 scores). M-Ratio denotes $\mathbf{M}$ is used only for performing mixup and sample selection is random. M-Threshold denotes that $\mathbf{M}$ is used to select samples based on the distance and mixup is performed with a random ratio.

We probe the individual impact of using matrix $\mathbf{M}$ for distance-based sample selection and using it for performing mixup in Table 4. We observe that both the applications of matrix $\mathbf{M}$ lead to improvements over TMix. Using matrix $\mathbf{M}$ for sample selection obtains larger improvements compared to using it as the ratio for performing mixup. This suggests that the selection of inputs for interpolation is more important than the mixing ratio when performing interpolative regularization.

4.4 Layer-wise Ablation

Mixup Layer Set	CoLA		HASOC		AHS	
	TMix	DMix	TMix	DMix	TMix	DMix
{3,4}	79.45	79.70	76.86	77.46	69.37	65.66
{0, 1, 2}	80.18	94.08	76.39	77.99	69.28	71.98
{6, 7, 9}	82.91	94.63	77.12	79.44	70.11	73.45
{7, 9, 12}	85.30	95.63	77.44	80.19	70.19	74.32
{3, 4, 6, 7, 9, 12}	84.03	**95.94**	76.99	**80.27**	70.03	**74.98**

Table 5: Layer-wise ablation (F1 scores) when performing interpolative augmentations.

We compare the performance of DMix and TMix for different sets of mixup layers in Table 5. TMix attains the best performance when the layer set $\{7, 9, 12\}$ is used since layers 6, 7, 9 and 12 contain the most amount of syntactic and semantic information (Chen et al., 2020a). Interestingly, DMix achieves the best performance when the layer is sampled from the set $\{3, 4, 6, 7, 9, 12\}$. This suggests that the surface-level information contained in layers 3 and 4 (Jawahar et al., 2019) is effectively leveraged by the distance-aware matrix M, leading to further improvements over purely syntactic and semantic information in layers $\{6, 7, 9, 12\}$.

4.5 Effect of Varying Thresholds

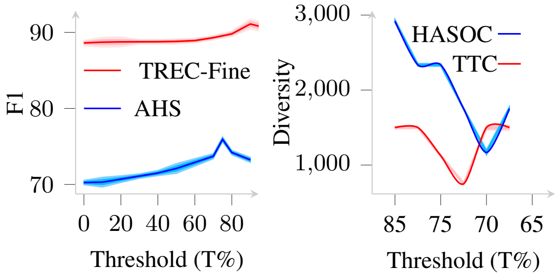


Figure 3: Change in performance in terms of F1 and Diversity with varying threshold T in % for DMix.

We perform a study by varying the threshold τ for DMix and present it in Figure 3. A decreasing τ denotes a larger distribution space for sampling instances for Mixup, and a T of 0% decomposes it to TMix or random Mixup. We observe an initial increase in the performance as we constrain the embedding space, suggesting the sampling of more diverse samples for interpolation. We observe a drop in performance when the constrain becomes very high, indicating that further expanding the sampling space does not lead to more diverse synthetic samples. This shows the existence of an optimum set of input samples for performing Mixup, and we conjecture it can be related to the sparsity in the embedding distribution of different languages.

5 Conclusion

We propose DMix, a novel data augmentation technique that interpolates samples intelligently chosen based on their hyperbolic distance in the embedding space. DMix achieves state-of-the-art results over existing data augmentation approaches on 8 standard and multilingual datasets in English, Arabic, Turkish, and Hindi languages, requiring 3 times less number of iterations than random mixup. DMix being independent of the underlying model and modality, holds potential to be applied on text, speech, and vision downstream tasks.

6 Acknowledgements

This work has been supported by the German Federal Ministry of Education and Research (BMBF) as a part of the Junior AI Scientists program under the reference 01-S20060. We thank the anonymous reviewers for their valuable inputs.

References

Nuha Albadi, Maram Kurdi, and Shivakant Mishra. 2018. Are they our brothers? analysis and detection of religious hate speech in the arabic twittersphere. In *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 69–76. ACM.

Shiladitya Bhattacharya, Siddharth Singh, Ritesh Kumar, Akanksha Bansal, Akash Bhagat, Yogesh Dawer, Bornini Lahiri, and Atul Kr. Ojha. 2020. Developing a multilingual annotated corpus of misogyny and aggression. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 158–168, Marseille, France. European Language Resources Association (ELRA).

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020a. MixText: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2147–2157, Online. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020b. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.

Octavian Ganea, Gary Becigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems*.

Raphael Gontijo-Lopes, Sylvia J Smullin, Ekin D Cubuk, and Ethan Dyer. 2020. Affinity and diversity: Quantifying mechanisms of data augmentation. *arXiv preprint arXiv:2002.08973*.

Hongyu Guo, Yongyi Mao, and Richong Zhang. 2019. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Amit Jindal, Arijit Ghosh Chowdhury, Aniket Didolkar, Di Jin, Ramit Sawhney, and Rajiv Ratn Shah. 2020. Augmenting NLP models using latent feature interpolations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6931–6936, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Deniz Kilinç, Akin Ozcift, Fatma Bozyiğit, Pelin Yildirim, Fatih Yucalar, and Emin Borandağ. 2017. Ttc-3600: A new benchmark dataset for turkish text categorization. *Journal of Information Science*, 43:174–185.

James P Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontañón. 2021. Fnet: Mixing tokens with fourier transforms. *ArXiv*, abs/2105.03824.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Linlin Liu, Bosheng Ding, Lidong Bing, Shafiq Joty, Luo Si, and Chunyan Miao. 2021. MulDA: A multilingual data augmentation framework for low-resource cross-lingual NER. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5834–5846, Online. Association for Computational Linguistics.

Thomas Mandl, Sandip Modha, Prasenjit Majumder, Daksh Patel, Mohana Dave, Chintak Mandlia, and Aditya Patel. 2019. Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, FIRE '19, page 14–17, New York, NY, USA. Association for Computing Machinery.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Linqing Shi, Danyang Liu, Gongshen Liu, and Kui Meng. 2020. Aug-bert: An efficient data augmentation algorithm for text classification. In *Communications, Signal Processing, and Systems*, pages 2191–2198, Singapore. Springer Singapore.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Alexandru Tifrea, Gary Becigneul, and Octavian-Eugen Ganea. 2019. Poincare glove: Hyperbolic word embeddings. In *International Conference on Learning Representations*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021. Entailment as few-shot learner. *ArXiv*, abs/2104.14690.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Shusheng Xu, Xingxing Zhang, Yi Wu, and Furu Wei. 2021. Sequence level contrastive learning for text summarization. *ArXiv*, abs/2109.03481.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Soyoung Yoon, Gyuwan Kim, and Kyumin Park. 2021. SSMix: Saliency-based span mixup for text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3225–3234, Online. Association for Computational Linguistics.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

A Extended Analysis

Model	**CoLA**	**TREC-Coarse**	**TREC-Fine**	**SST-2**
XLNet (2019)	70.20	94.58	87.49	**97.00**
T5-small (2020)	71.60	95.55	86.21	91.80
FNet (2021)	78.00	96.89	89.97	94.00
EFL (2021)	86.40	93.36	80.90	96.90
EMix (2020)	85.21	97.44	90.04	91.13
SSMix (2021)	86.76	97.60	90.24	92.95
DMix (Ours)	**95.94**	**97.80**	**91.14**	92.44

Table 6: Performance comparison with additional baselines and interpolative augmentation methods.

We compare the performance of DMix on standard English and GLUE datasets with additional baselines and interpolative augmentation methods like EMix (Jindal et al., 2020) and SSMix (Yoon et al., 2021).

B Dataset Details

1. **TRAC**. (Bhattacharya et al., 2020) is a collection of posts, comments, and other content from popular social media, streaming and sharing platforms. For the purpose of our experiments, we perform the aggression classification task, for which, the data is labelled into 3 classes based on the level of aggression.

2. **TREC-Coarse**. (Li and Roth, 2002), The Text REtrieval Conference-Coarse is a question classification dataset consisting of 6 classes. The data is sourced from English questions by USC, TREC 8, TREC 9, TREC 10 and manually constructed questions.

3. **TREC-Fine**. (Li and Roth, 2002) contains the same set of questions as TREC-Coarse grouped into 47 fine-grained classes instead of 6.

4. **CoLA**. (Warstadt et al., 2018), abbreviation for the Corpus of Linguistic Acceptability is a part of GLUE (Wang et al., 2018) benchmark. It is a collection of English sentences from 23 linguistic publications that are annotated for their grammatical acceptability.

5. **SST-2**. (Socher et al., 2013) is a GLUE (Wang et al., 2018) benchmark dataset consisting of English sentences from movie reviews. Samples in the dataset are annotated for sentiment classification task.

6. **AHS**. (Albadi et al., 2018) is an Arabic hate speech classification dataset focusing mainly on Saudi Twittersphere. The data has been collected over a span of 6 months from March 2018 to August 2018 and has 3950 samples classified into 2 classes.

7. **TTC**. (Kilinç et al., 2017), Turkish Text Categorization dataset consists of 3600 Turkish documents (news/texts) classified into 6 classes. The data is obtained between the period from May 2015 to July 2015.

8. **HASOC**. (Mandl et al., 2019) consists of content sampled from social media platforms. We perform the binary Hate/Offensive content classification task on the Hindi dataset for the purpose of our experiments.

C Experimental Setup

We mention the optimal hyperparameter settings in Table 8.

Sentence	TMix	DMix-NT	DMix
Intellectuals and the so-called Secular are more illiterate Uneducated and illiterate	OAG	NAG	NAG
She must be sent to jail for anti national activities under NSA and PSA	NAG	CAG	CAG
Lion king fan hit like	OAG	CAG	NAG
kapil why are u listening to these ch∗tsss ....give them shut up call...insane idiots	CAG	CAG	OAG
Great Job Mr Jahangir Sir I support you	NAG	CAG	NAG
Absolute fantastic movie please go and watch the movie first.	CAG	NAG	NAG

Table 7: Qualitative analysis of the performance obtained by TMix, DMix-NT, and DMix. The color intensity of each word corresponds to the token-level attention score. Green denotes correct prediction and red denotes incorrect prediction. (NAG: Non Aggressive, OAG: Overtly Aggressive, CAG: Covertly Aggressive).

Parameter	Value
Optimizer	BERTAdam
Learning Rate	2e-5
Batch Size	8
$\beta_1, \beta_2, \epsilon$	0.9, 0.999, 1e-6
# Epochs	5
Evaluation Metric	F1 Score
Base Model	BERT-base-uncased, BERT-base-multilingual-uncased
Classifier (over architecture)	Linear layer
Hardware	Nvidia P100

Table 8: Model and training setup for DMix.

D Comparison with Contrastive Learning

Contrastive learning involves training the underlying model to learn an embedding space in which similar sample pairs stay close to each other while dissimilar ones are far apart. Hence, their training objective directly involves training using this embedding vector of the input samples in the dataset. DMix however chooses samples based on their spatial distribution in the embedding space, but does not have a training objective optimizing on their position in the embedding space. The training of DMix is still supervised in nature and involves learning over the mixed label of the individual samples being used for interpolation.

E Qualitative Analysis

To further analyze DMix, we perform a qualitative study by choosing examples from the dataset and compare the predictions made by TMix and DMix-NT with DMix. We analyze token-level attention assigned to the individual terms by BERT, where color intensity corresponds to the attention score. We present these results in Table 7.

Sub-Word Alignment is Still Useful: A Vest-Pocket Method for Enhancing Low-Resource Machine Translation

Minhan Xu, Yu Hong*
School of Computer Science and Technology, Soochow University, China
cosmosbreak5712@gmail.com, tianxianer@gmail.com

Abstract

We leverage embedding duplication between aligned sub-words to extend the Parent-Child transfer learning method, so as to improve low-resource machine translation. We conduct experiments on benchmark datasets of My→En, Id→En and Tr→En translation scenarios. The test results show that our method produces substantial improvements, achieving the BLEU scores of 22.5, 28.0 and 18.1 respectively. In addition, the method is computationally efficient which reduces the consumption of training time by 63.8%, reaching the duration of 1.6 hours when training on a Tesla 16GB P100 GPU. All the models and source codes in the experiments will be made publicly available to support reproducible research.

1 Introduction

Low-resource machine translation (MT) is challenging due to the scarcity of parallel data and, in some cases, the absence of bilingual dictionaries (Zoph et al., 2016; Miceli Barone, 2016; Koehn and Knowles, 2017; Zhang et al., 2017). Unsupervised, multilingual and transfer learning have been proven effective in the low-resource MT tasks, grounded on different advantages (section 2).

In this paper, we follow Aji et al. (2020)'s work to utilize cross-language transfer learning, of which the "parent-child" transfer framework is first proposed by Zoph et al. (2016). In the parent-child scenario, a parent MT model and a child MT model are formed successively, using the same neural network structure. In order to achieve the sufficient warm-up effect from scratch, the **parent** is trained on **high**-resource language pairs. Further, the **child** inherits the parent's properties (e.g., inner parameters and embedding layers), and it is boosted by the fine-tuning over **low**-resource language pairs. One of the distinctive contributions in Aji et al. (2020)'s study is to demonstrate the significant effect of embedding duplication for transference, when it is conducted between the morphologically-identical sub-words in different languages.

We attempt to extend Aji et al. (2020)'s work by additionally duplicating embedding information among the aligned multilingual sub-words. It is motivated by the assumption that if the duplication between morphologically-identical sub-words contributes to cross-language transference, the duplication among any other type of equivalents is beneficial in the same way, such as that of the aligned sub-words, most of which are likely to be morphologically-dissimilar but semantically-similar (or even exactly the same).

In our experiments, both the parent and child MT models are built with the transformer-based (Vaswani et al., 2017) encoder-decoder architecture (Section 3.1). We use the unigram model from SentencePiece (Kudo and Richardson, 2018) for tokenizing, and carry out sub-word alignment using eflomal (Section 3.2). On the basis, we develop a normalized element-wise embedding aggregation method to tackle the many-to-one embedding duplication for aligned sub-words (Section 3.3). The experiments show that our method achieves substantial improvements without using data augmentation.

2 Related Work

The majority of previous studies can be sorted into 3 aspects in terms of the exploited learning strategies, including unsupervised, multilingual and transfer learning.

- **Unsupervised** MT conducts translation merely conditioned on monolingual language models (Lample et al., 2018a; Artetxe et al., 2017). The ingenious method that has been explored successfully is to bridge the source and target languages using a shareable

[*] Corresponding author.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 613 - 619
May 22-27, 2022 ©2022 Association for Computational Linguistics

representation space (Lampple et al., 2018b), which is also known as interlingual (Cheng et al., 2017) or cross-language embedding space (Kim et al., 2018). To systematize unsupervised MT, most (although not all) of the arts leverage bilingual dictionary induction (Conneau et al., 2018; Søgaard et al., 2018), iterative back-translation (Sennrich et al., 2016a; Lample et al., 2018b) and denoised auto-encoding (Vincent et al., 2008; Kim et al., 2018).

- **Multilingual** MT conducts translation merely using a single neural model whose parameters are thoroughly shared by multiple language pairs (Firat et al., 2016; Lee et al., 2017; Johnson et al., 2017; Gu et al., 2018a,b), including a variety of high-resource language pairs as well as a kind of low-resource (the target language is fixed and definite). Training on a mix of high-resource and low-resource (even zero-resource) language pairs enables the shareable model to generalize across language boundaries (Johnson et al., 2017). The benefits result from the assimilation of relatively extensive translation experience and sophisticated modes from high-resource language pairs.

- **Transferable** MT is fundamentally similar to multilingual MT, whereas it tends to play the aforementioned Parent-Child game (Zoph et al., 2016). A variety of optimization methods have been proposed, including the transfer learning over the embeddings of WordPieces tokens (Johnson et al., 2017), BPE sub-words (Nguyen and Chiang, 2017) and the shared multilingual vocabularies (Kocmi and Bojar, 2018; Gheini and May, 2019), as well as the transference that is based on the artificial or automatic selection of congeneric parent language pairs (Dabre et al., 2017; Lin et al., 2019). In addition, Aji et al. (2020) verify the different effects of various transferring strategies of sub-word embeddings, such as that among morphologically-identical sub-words.

In this paper, we extend Aji et al. (2020)'s work, transferring embedding information not only among the morphologically-identical sub-words but the elaborately-aligned sub-words.

3 Approach

3.1 Preliminary: Basic Transferable NMT

We follow Kim et al. (2019) and Aji et al. (2020) to build neural MT (NMT) models with 12-layer transformers (Vaswani et al., 2017), in which the first 6 layers are used as the encoder while the subsequent 6 layers the decoder.

Embedding Layer As usual, the encoder is coupled with a trainable embedding layer, which maintains a fixed bilingual vocabulary and trainable sub-word embeddings. Each embedding is specified as a 512-dimensional real-valued vector.

Parent-Child Transfer We follow Zoph et al. (2016) to conduct Parent-Child transfer learning. Specifically, we adopt an off-the-shelf transformer-based NMT[1] which was adequately trained on high-resource De→En (German→English) language pairs. The publicly-available data of OPUS (Tiedemann, 2012) is used for training, which comprises about 351.7M De→En parallel sentence pairs[2]. We regard this NMT model as the Parent. Further, we transfer all inner parameters of the 12-layer transformers from Parent to Child.

By contrast, the embedding layer of Parent is partially transferred to Child, which has been proven effective in Aji et al. (2020)'s study. Assume V_h denotes the high-resource (e.g., the aforementioned De-En) vocabulary while V_l the low-resource, the morphologically-identical sub-words V_o are then specified as the ones occurring in both V_h and V_l (i.e., $V_o = V_h \cap V_l$). Thus, we duplicate the embeddings of morphologically-identical sub-words V_o from the embedding layer of Parent to that of Child. Further, we randomly initialize the embeddings of the rest sub-words V_r in the Child's embedding layer ($V_r = V_l - V_o$), where random sampling from a Gaussian distribution is used.

Both the transferred inner parameters and the duplicated embeddings constitutes the initial state of the Child NMT model. On the basis, we fine-tune Child on the low-resource language pairs, such as the considered 18K My→En (Burmese→English) parallel data in our experiments.

3.2 Tokenizer and Alignment

We strengthen Parent-Child transfer learning by additionally duplicating embeddings for aligned sub-words (between low and high-resource languages).

[1] https://github.com/Helsinki-NLP/OPUS-MT-train/blob/master/models/de-en/README.md

[2] https://opus.nlpl.eu/

	Doc.	Sent.	Token
My	113K	1.1M	17.4M
Id	1.1M	8.3M	156.2M
Tr	705K	5.8M	128.2M

Table 1: Statistics of monolingual Wikipedia data.

	Train.	Val.	Test
My-En (ALT)	18K	1K	1K
Id-En (BPPT)	22K	1K	1K
Tr-En (WMT17)	207K	3K	3K

Table 2: Statistics for low-resource parallel datasets.

The precondition is to produce the word-level alignment and equivalently assign it to sub-words.

Word Alignment We use Eflomal[3] to achieve the word alignment. It is developed based on EFMARAL (Östling et al., 2016), where Gibbs sampling is run for inference on Bayesian HMM models. Eflomal is not only computationally efficient but able to perform n-to-1 alignment. We separately train Eflomal on the low-resource My→En, Id (Indonesian)→En and Tr (Turkish)→En parallel data (Section 4).

Sub-word Tokenizer We train a sub-word tokenizer using the unigram model of SentencePiece for each low-resource language, including My, Id and Tr. The tokenizers are trained on monolingual plain texts which are collected from Wikipedia's dumps[4]. The toolkit wikiextractor[5] is utilized to extract plain texts from the semi-structured data. The statistics of training data is shown in Table 1.

We uniformly set the size of sub-word vocabulary to 50K when training the tokenizers. The obtained vocabulary of each low-resource language is utilized for sub-word alignment, towards the mixed De-En sub-word vocabulary in the Parent NMT model. The size of De-En vocabulary is 58K.

Sub-word Alignment Given a pair of aligned bilingual words, we construct the same correspondence for their sub-words by many-to-many mappings. See the De→Tr example in (1).

(1) Word Alignment: | *produktion↔üretme*
| *Harnstoff↔üre*
Sub-word Alignment: | *produck*↔{***üre***, *tme*}
| *tion*↔{***üre***, *tme*}
| *Harn*↔{***üre***}
| *stoff*↔{***üre***}

It is unavoidable that some of the aligned sub-words are non-canonical. Though, the positive effect on transfer learning may be more substantial than negative. It motivated by the findings that the use of sub-words ensures a sufficient overlap between vocabularies (Nguyen and Chiang, 2017), and thus enables the transfer of a larger number of concrete embeddings rather than random ones.

3.3 N-to-1 Embedding Duplication

Assume that V_l^a denotes the sub-words in low-resource vocabulary that have aligned sub-words in high-resource vocabulary, the mapping is $D(x)$, note that $\forall x \in V_l^a$, $D(x)$ is a set of sub-words. Thus, in the embedding layer of Child, we extend the range of sub-words for embedding transfer, including both the identical sub-words V_o and the aligned V_l^a. To enable the transfer, we tackle n-to-1 embedding duplication. It is because that, in a large number of cases, there is more than one high-resource sub-word corresponding to a single low-resource sub-word (see "*üre*" in (1)).

Given a sub-word x in V_l^a and the aligned sub-words v_x in $D(x)$, we rank v_x in terms of the frequency with which they were found to be aligned with x in the parallel data. On the basis, we carry out two duplication methods as below.

- **Top-1** We take the top-1 sub-word $\check{x}$ from v_x, and perform element-wise embedding duplication from $\check{x}$ to x: $\forall i, E_i(\check{x}) = E_i(x)$ (i is the i-th dimension of embedding $E(*)$).

- **Mean** We adopt all the sub-words in v_x, and duplicate their embedding information by the normalized element-wise aggregation (where, n denotes the number of sub-words in v_x):

$$\forall i, E_i(\check{x}) = \sum_{x \in v_x} E_i(x)/n$$

4 Experimentation

4.1 Datasets and Evaluation Metric

We evaluate the transferable NMT models for three source languages (My, Id and Tr). English is invariably specified as the target language. There are three low-resource parallel datasets used for training the Child NMT model, including Asian Language Treebank (ALT) (Ding et al., 2018), PAN Localization BPPT[6] and the corpus of WMT17 news

[3] https://github.com/robertostling/eflomal

[4] https://dumps.wikimedia.org

[5] https://github.com/attardi/wikiextractor

[6] http://www.panl10n.net/english/OutputsIndonesia2.htm

Model	**My-En**	**Id-En**	**Tr-En**
Baseline	20.5	26.0	17.0
MI-PC	21.0	27.5	17.6
Top-1-PC	21.9	27.6	18.0
Mean-PC	**22.5**	**28.0**	**18.1**

Table 3: Results using **SentencePiece** tokenizer.

Model	**My-En**	**Id-En**	**Tr-En**
Baseline	20.2	24.5	16.5
MI-PC	20.4	24.2	16.8
Top-1-PC	21.2	26.9	16.9
Mean-PC	**21.9**	**27.1**	**16.9**

Table 4: Results using **BPE** tokenizer.

translation task (Bojar et al., 2017). The statistics in the training, validation and test sets is shown in Table 2. We evaluate all the considered NMT models with SacreBLEU (Post, 2018).

4.2 Hyperparameters

We use an off-the-shelf NMT model as Parent (Section 3.1), whose state variables (i.e., hyperparameters and transformer parameters) and embedding layer are all set. On the contrary, the Child NMT model needs to be regulated from scratch.

When training and developing Child, we adopt the following hyperparameters. Each source language was tokenized using SentencePiece (Kudo and Richardson, 2018) with 50k vocabulary size. Training was carried out with HuggingFace Transformers library (Wolf et al., 2020) using the Adam optimizer with 0.1 weight decay rate. The maximum sentence length was set to 128 and the batch size to 64 sentences. The learning rate was set to 5e-5 and checkpoint frequency to 500 updates. For each model, we selected the checkpoint with the lowest perplexity on the validation set for testing.

5 Results and Analysis

Table 3 shows the test results, where all the considered Parent-Child transfer models are marked with "PC", and the baseline is the transformer-based NMT (Section 3.1) which is trained merely using low-resource parallel data (without transfer learning). MI-PC is the reproduced transfer model in terms of Aji et al. (2020)'s study, in which only the embedding transference of morphologically-identical sub-words is used. We report NMT performance when MI-PC is used to enhance the baseline, as well as that when our auxiliary transfer models (i.e., Top-1 and Mean in Section 3.3) are additionally adopted, separately.

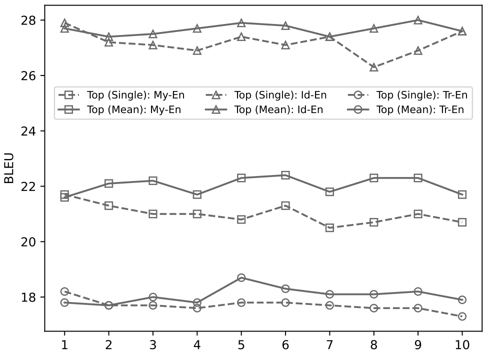


Figure 1: Comparison between embedding duplication of a single aligned sub-word (denoted with **Single**) and that of multiple sub-words (**Mean**).

Model	**My-En**	**Id-En**	**Tr-En**
Baseline	1.30	1.27	4.49
MI-PC	1.30	1.35	3.53
Top-1-PC	1.11	1.00	3.07
Mean-PC	**0.96**	**0.94**	**2.14**

Table 5: The time (in hour) that different MT models consumed during training in all experiments (0.9 hour is equivalent to 54 minutes).

It can be observed that, compared to MI-PC, both Top-1-PC and Mean-PC yield improvements for all the three low-resource MT scenarios. The most significant improvement occurs for My→En MT, reaching up to 1.5 BLEU. Both the models generalize well across changes in the input sub-words. It can be illustrated in a separate experiment where the BPE (Sennrich et al., 2016b) tokenizer is used (instead of SentencePiece (Kudo and Richardson, 2018)), and all the transfer models are run over the newly-aligned sub-words. As shown in Table 4, both Top-1-PC and Mean-PC still outperform MI-PC, yielding an improvement of 2.9 BLEU at best (for Id→En MT).

Due to unavoidable errors in the sub-word alignment, the utilization of a single aligned sub-word for embedding duplication easily results in performance degradation. Aggregating and normalizing embeddings of all possible aligned sub-words help to overcome the problem. Figure 1 shows the NMT performance obtained when the i-th top-ranked aligned sub-word is exclusively used for transfer, as well as the aggregation of top-i sub-words is used. It can be found that the latter model almost always outperforms the former model.

We compare the training time consumption of all experiments, the result is shown in Table 5. We use mixed precision for training the child MT model. All experiments are conducted on a single NVIDIA P100 16GB GPU.

Obviously, the time that Mean-PC consumes during training is less than other models. In the scenario of Tr-En MT, the training duration is even shortened from 4.49 hours (i.e., about 269 minutes) to 2.14, compared to the baseline model. Most probably, it is caused by the transferring of a larger number of sub-word embeddings during training. In other word, Mean-PC actually transfers not only morphologically-identical sub-words but the aligned ones. This contributes more to the avoidance of redundant learning over sub-word embeddings. All in all, Mean-PC is less time-consuming when producing substantial improvements.

6 Conclusion

We enhance transferable Parent-Child NMT by duplicating embeddings of aligned sub-words. The experimental results demonstrate that the proposed method yields substantial improvements for all the considered MT scenarios (including My-En, Id-En and Tr-En). More importantly, we successfully reduce the training duration. The efficiency can be improved with the ratio of about 50% at best.

Additional survey in the experiments reveals that phonetic symbols can be used for transfer learning between the languages belonging to different families. For example, the phonologies of *hamburger* in German and Burmese are similar (Há́mburger vs hambhargar). In the future, we will study bilingual embedding transfer of phonologically-similar words, so as to further improve low-resource NMT.

Acknowledgements

The research is supported by National Key R&D Program of China (2020YFB1313601) and National Science Foundation of China (62076174).

References

Alham Fikri Aji, Nikolay Bogoychev, Kenneth Heafield, and Rico Sennrich. 2020. In neural machine translation, what does transfer learning transfer? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7701–7710, Online. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

Yong Cheng, Yang Liu, Qian Yang, Maosong Sun, and Wei Xu. 2017. Joint training for pivot-based neural machine translation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 3974–3980.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *Proceedings of the 6th International Conference on Learning Representations*.

Raj Dabre, Tetsuji Nakagawa, and Hideto Kazawa. 2017. An empirical study of language relatedness for transfer learning in neural machine translation. In *Proceedings of the 31st Pacific Asia Conference on Language, Information and Computation*, pages 282–286. The National University (Phillippines).

Chenchen Ding, Masao Utiyama, and Eiichiro Sumita. 2018. Nova: A feasible and flexible annotation system for joint tokenization and part-of-speech tagging. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 18(2):1–18.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 866–875, San Diego, California. Association for Computational Linguistics.

Mozhdeh Gheini and Jonathan May. 2019. A universal parent model for low-resource neural machine translation transfer. *arXiv preprint arXiv:1909.06516*.

Jiatao Gu, Hany Hassan, Jacob Devlin, and Victor O.K. Li. 2018a. Universal neural machine translation for extremely low resource languages. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 344–354, New Orleans, Louisiana. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018b. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods* *in Natural Language Processing*, pages 3622–3631, Brussels, Belgium. Association for Computational Linguistics.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Yunsu Kim, Yingbo Gao, and Hermann Ney. 2019. Effective cross-lingual transfer of neural machine translation models without shared vocabularies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1246–1257, Florence, Italy. Association for Computational Linguistics.

Yunsu Kim, Jiahui Geng, and Hermann Ney. 2018. Improving unsupervised word-by-word translation with language model and denoising autoencoder. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 862–868, Brussels, Belgium. Association for Computational Linguistics.

Tom Kocmi and Ondřej Bojar. 2018. Trivial transfer learning for low-resource neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 244–252, Brussels, Belgium. Association for Computational Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *Proceedings of the 6th International Conference on Learning Representations*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018b. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, Brussels, Belgium. Association for Computational Linguistics.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastasopoulos, Patrick Littell, and Graham Neubig. 2019. Choosing transfer languages for cross-lingual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy. Association for Computational Linguistics.

Antonio Valerio Miceli Barone. 2016. Towards cross-lingual distributed representations without parallel text trained with adversarial autoencoders. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 121–126, Berlin, Germany. Association for Computational Linguistics.

Toan Q. Nguyen and David Chiang. 2017. Transfer learning across low-resource, related languages for neural machine translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 296–301, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Robert Östling, Jörg Tiedemann, et al. 2016. Efficient word alignment with markov chain monte carlo. *The Prague Bulletin of Mathematical Linguistics*.

Matt Post. 2018. A call for clarity in reporting bleu scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. 2018. On the limitations of unsupervised bilingual dictionary induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 778–788, Melbourne, Australia. Association for Computational Linguistics.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

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

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1970, Vancouver, Canada. Association for Computational Linguistics.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

HYPHEN: Hyperbolic Hawkes Attention For Text Streams

Shivam Agarwal[*], **Ramit Sawhney**[*], **Sanchit Ahuja, Ritesh Soun, Sudheer Chava**
Financial Services Innovation Lab, Georgia Institute of Technology
rsawhney31@gatech.edu, sudheer.chava@scheller.gatech.edu

Abstract

Analyzing the temporal sequence of texts from sources such as social media, news, and parliamentary debates is a challenging problem as it exhibits time-varying scale-free properties and fine-grained timing irregularities. We propose a Hyperbolic Hawkes Attention Network (HYPHEN), which learns a data-driven hyperbolic space and models irregular powerlaw excitations using a hyperbolic Hawkes process. Through quantitative and exploratory experiments over financial NLP, suicide ideation detection, and political debate analysis we demonstrate HYPHEN's practical applicability for modeling online text sequences in a geometry agnostic manner.

1 Introduction

Text stream modeling is a critical problem that helps analyze trends over a variety of applications spanning finance (Oliveira et al., 2017), healthcare (Baytas et al., 2017), and political discourses (Sawhney et al., 2021c). However, analyzing such text sequences poses several challenges. First, modeling individual text items may not be informative enough since text sequences display a *sequential context dependency*, where analyzing them together in succession provides better contextual representation (Hu et al., 2018). Second, timing plays an essential role in online stream modeling as users quickly react to new information (Sawhney et al., 2021a). For instance, in stock markets, reacting a second slower than other investors can lead to massive losses (Scholtus et al., 2014). A fundamental limitation in existing RNN methods is that it ignores the natural fine-grained timing irregularities in streams (Foucault et al., 2016; Eysenck, 1968).

Social theories show that from a vast volume of texts in a stream, only a few are powerful enough to heavily influence the overall trend (Van Dijk, 1977; Gabaix, 2016). Such texts are rare and the excitation induced by them follows a powerlaw distribution which gives rise to scale-free properties (Zhao et al., 2010). For example, in political debates, there are a few rare highly-influential debates that heavily impact the overall voting decisions of citizens (Law, 2019). Further, the impact of such powerlaw excitations varies for each event. The presence of varying powerlaw dynamics from highly influential texts correlates with natural hierarchies and scale-free dynamics in text streams, making them difficult to model (Sala et al., 2018).

The good news is that hyperbolic learning has shown to better model such powerlaw dynamics compared to Euclidean learning over domains, including vision (Khrulkov et al., 2020) and NLP (Tifrea et al., 2019). However, existing works face two major limitations, 1) they ignore the timing irregularities in scale-free sequences and 2) they use a single hyperbolic space to encode varying levels of hyperbolic dynamics. Building on social theories, our contributions can be summarized as:

- We explore the hyperbolic properties of online streams and propose a Hyperbolic Hawkes Attention Network (HYPHEN) which jointly learns from the fine-grained timing irregularities and powerlaw dynamics of streams (§**2.2**).
- Building on social theories, HYPHEN learns the hyperbolic space based on the nature of the stream (§**2.1**). We introduce HYPHEN as a geometry agnostic model which can be applied on any downstream application.
- Through quantitative (§**4.1**) and exploratory (§**4.3**) experiments on four tasks spanning suicide ideation, political debate analysis, and financial forecasting over English and Chinese languages, we demonstrate the practical applicability of HYPHEN for stream modeling.[1]

[*] Equal contribution.

[1] We release HYPHEN's code at: https://github.com/gtfintechlab/HYPHEN-ACL

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 620 - 627
May 22-27, 2022 ©2022 Association for Computational Linguistics

2 Methodology

Problem Formulation: For a sequence of texts $[p_1 \dots, p_N]$ released at times $[t_1, \dots, t_N]$ sequentially, with $[t_1 < \dots < t_N]$, our target is to model this sequence in a time-sensitive fashion for a variety of downstream applications (§3).

2.1 Learnable Hyperbolic Geometry

Text sequences from social media and political discourses pose hierarchies (Sawhney et al., 2021a) i.e., the datasets represent a tree like structure which call for the use of hyperbolic spaces. Indeed, the volume of hyperbolic geometry grows exponentially, in contrast to Euclidean spaces where the growth is polynomial (Khrulkov et al., 2020), enabling hyperbolic spaces to capture the underlying scale-free properties of streams (Sala et al., 2018). However, text sequences exhibit a varying degree of scale-free dynamics, which a single geometry cannot capture (Gu et al., 2019). Thus, we seek to learn the optimal underlying geometry.

The hyperbolic space is a non-Euclidean space with a constant negative curvature c. To learn the optimal geometry, we aim to learn the curvature c, which controls the degree of hyperbolic properties represented by the space (Gu et al., 2019). Following (Ganea et al., 2018) we define the hyperbolic geometry with varying curvature c as $(\mathcal{B}, g_x^{\mathcal{B}})$, where the manifold $\mathcal{B} = \{x \in \mathbb{R}^n : c||x|| < 1\}$, is endowed with the Riemannian metric $g_x^{\mathcal{B}} = \lambda_x^2 g^E$, where the conformal factor $\lambda_x = \frac{2}{1-c||x||^2}$ and $g^E = \text{diag}[1, .., 1]$ is the Euclidean metric tensor. We denote the tangent space centered at point $\boldsymbol{x}$ as $\mathcal{T}_{\boldsymbol{x}}\mathcal{B}$. We generalize Euclidean operations to the hyperbolic space via Möbius operations.

Möbius Addition $\oplus$ for two points $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{B}$, is,

$$x \oplus y = \frac{(1 + 2c\langle x, y\rangle + c||y||^2)x + (1 - c||x||^2)y}{1 + 2c\langle x, y\rangle + c^2||x||^2||y||^2} \quad (1)$$

$\langle ., . \rangle$, $|| \cdot ||$ denotes the inner product and norm.

Exponential Map maps a tangent vector $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathcal{B}$ to a point $\exp_{\boldsymbol{x}}(\boldsymbol{v})$ in the hyperbolic space,

$$\exp_{\boldsymbol{x}}(\boldsymbol{v}) = \boldsymbol{x} \oplus \left(\tanh\left(\frac{\sqrt{c}\lambda_{\boldsymbol{x}}||\boldsymbol{v}||}{2}\right)\frac{\boldsymbol{v}}{\sqrt{c}||\boldsymbol{v}||}\right) \quad (2)$$

Logarithmic Map maps a point $\boldsymbol{y} \in \mathcal{B}$ to a point $\log_{\boldsymbol{x}}(\boldsymbol{y})$ on the tangent space at $\boldsymbol{x}$,

$$\log_{\boldsymbol{x}}(\boldsymbol{y}) = \frac{2}{\sqrt{c}\lambda_{\boldsymbol{x}}}\tanh^{-1}\left(\sqrt{c}||-\boldsymbol{x}\oplus\boldsymbol{y}||\right)\frac{-\boldsymbol{x}\oplus\boldsymbol{y}}{||-\boldsymbol{x}\oplus\boldsymbol{y}||} \quad (3)$$

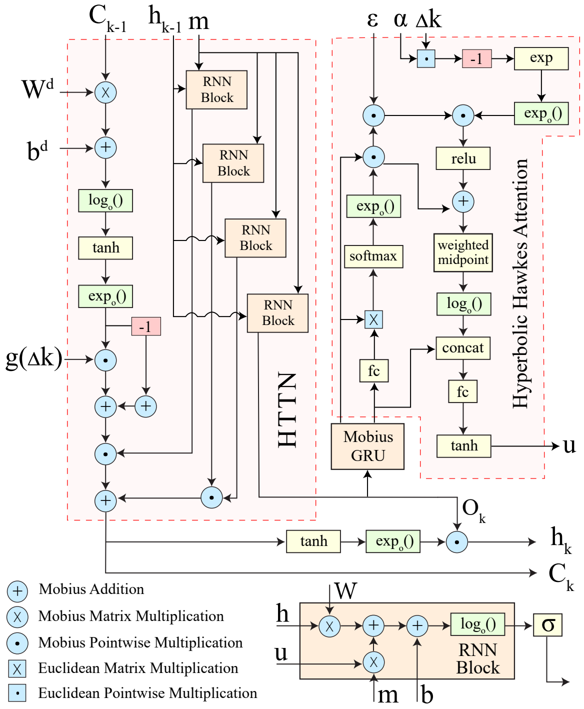


Figure 1: HYPHEN cell diagram and update rule.

Möbius Multiplication $\otimes$ multiplies features $\boldsymbol{x} \in \mathcal{B}^C$ with matrix $\boldsymbol{W} \in \mathbb{R}^{C' \times C}$, given by

$$\boldsymbol{W} \otimes \boldsymbol{x} = \exp_{\boldsymbol{o}}(\boldsymbol{W}\log_{\boldsymbol{o}}(\boldsymbol{x})) \quad (4)$$

Möbius Pointwise Product $\odot$ multiplies matrix $\boldsymbol{x} \in \mathcal{B}^C$ with matrix $\boldsymbol{y} \in \mathcal{B}^C$ pointwise,

$$\boldsymbol{x} \odot \boldsymbol{y} = \frac{1}{\sqrt{c}}\tanh\left(\frac{||xy||}{y}\arctan^{-1}(\sqrt{c}||y||)\right)\frac{||xy||}{||y||} \quad (5)$$

2.2 HYPHEN: Hyperbolic Hawkes Network

Text Embedding Layer We use Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) to encode each text p_i to features $\hat{\boldsymbol{m}}_{\boldsymbol{i}} = \text{BERT}(p_i) \in \mathbb{R}^d$ where $d = 768$, obtained by averaging the token level outputs from the final layer of BERT. To apply hyperbolic operations over text features $\hat{\boldsymbol{m}}_{\boldsymbol{i}}$, we project it to the hyperbolic space via the exponential mapping $\exp_o(\cdot)$ given by, $\boldsymbol{m}_{\boldsymbol{i}} = \exp_o(\hat{\boldsymbol{m}}_{\boldsymbol{i}})$

Hyperbolic Time Aware Temporal Network To encode the varying scale-free characteristics of text sequences, we introduce LSTMs over learnable hyperbolic spaces by leveraging Möbius operations (§2.1). Further, capturing fine-grained timing irregularities in text streams plays a crucial role for stream state modeling. For instance, the time interval between two debates can vary widely, from a few days to many months in parliamentary debates. Consequently, the ideologies and thought process of the speaker may change over time, reflecting a decay or increase in dependence on the speaker's previous speeches (Van Dijk, 2002).

To capture these time dependent intricacies in a learnable hyperbolic space, we modify the hyperbolic LSTM (Shimizu et al., 2021) as shown in Figure 1 into a hyperbolic time-aware temporal network (HTTN$(\cdot)$). Intuitively, the greater the time elapsed between text releases, the lesser the impact they should have on each other. Thus, for a given day k, HTTN applies a decaying function over Δk, the elapsed time between two texts $[p_k, p_{k-1}]$, transforming the time differences into weights:

$$\begin{aligned}
\boldsymbol{C}^s_{k-1} &= \exp_{\boldsymbol{o}}(\tanh(\log_{\boldsymbol{o}}(\boldsymbol{W}^d \otimes \boldsymbol{C}_{k-1} \oplus \boldsymbol{b}^d))) \\
\hat{\boldsymbol{C}}^s_{k-1} &= \boldsymbol{C}^s_{k-1} \odot g(\Delta k) \quad \text{Discounted short-term memory} \\
\boldsymbol{C}^T_{k-1} &= -\boldsymbol{C}^s_{k-1} \oplus \boldsymbol{C}_{k-1} \qquad \text{Long term memory} \\
\boldsymbol{C}^*_{k-1} &= \boldsymbol{C}^T_{k-1} \oplus \hat{\boldsymbol{C}}^s_{k-1} \qquad \text{Adjusted previous memory}
\end{aligned}$$

where $\boldsymbol{C}^s_{k-1}$ is the previous cell memory, $\boldsymbol{W}^d; \boldsymbol{b}^d$ are the network parameters, and $g(\cdot)$ is a heuristic decaying function. Following (Baytas et al., 2017) we set $g(\Delta k) = 1/\Delta k$. Using the adjusted previous memory $\boldsymbol{C}^*_{k-1}$, we define the current hidden state and current memory states for HTTN, with hyperbolic features m as:

$$\begin{aligned}
\widetilde{\boldsymbol{c}}_{\boldsymbol{k}} &= \sigma \log_{\boldsymbol{o}}(\boldsymbol{W}^c \otimes \boldsymbol{h}_{k-1} \oplus \boldsymbol{U}^c \otimes \boldsymbol{m}_k \oplus \boldsymbol{b}^c) \\
\boldsymbol{C}_k &= \boldsymbol{i}_k \odot \widetilde{\boldsymbol{c}}_{\boldsymbol{k}} \oplus \boldsymbol{f}_k \odot \boldsymbol{C}^*_{k-1} \qquad \text{(Current memory)} \\
\boldsymbol{h}_k &= \boldsymbol{o}_k \odot \exp_{\boldsymbol{o}}(\tanh(\boldsymbol{C}_k)) \qquad \text{(Current hidden state)}
\end{aligned}$$

where $\boldsymbol{W}^c; \boldsymbol{U}^c; \boldsymbol{b}^c$ are the learnable parameters, $\boldsymbol{i}_k; \boldsymbol{f}_k; \boldsymbol{o}_k$ are input, forget and output gates. Finally, given texts $[p_1, \dots p_T]$ over a lookback period T, we define the update rule of HTTN as,

$$\boldsymbol{h_j} = \text{HTTN}(\boldsymbol{m_j}, \Delta j, \boldsymbol{h_{j-1}}); \quad j \in [1, T] \tag{6}$$

where, h_j represents the hidden states of HTTN.

Hyperbolic Hawkes Attention Studies show that not all historical texts are equally informative and pose a *diverse influence* over the predictions (Sawhney et al., 2021c). We use a temporal hyperbolic attention mechanism (Luong et al., 2015) to emphasize texts likely to have a substantial influence. This mechanism learns attention weights β_i for each hiddden state $\boldsymbol{h_i} \in \overline{\boldsymbol{h}} = [\boldsymbol{h_1}, \dots, \boldsymbol{h_T}]$ as,

$$\beta_j = \text{Softmax}\Big(\exp\left(\log_{\boldsymbol{o}}(\boldsymbol{h_j})^{\mathrm{T}}(\boldsymbol{W} \log_{\boldsymbol{o}}(\overline{\boldsymbol{h}}))\right)\Big) \tag{7}$$

where, $\boldsymbol{W}$ denotes learnable weights.

Next, we enhance the temporal hyperbolic attention using the Hawkes process (Mei and Eisner, 2017) and propose a hyperbolic Hawkes attention mechanism. The Hawkes process is a temporal point process that models a sequence of arrival of texts over time. Each text item *"excites"* the process in the sense that the chance of a subsequent arrival is increased for some time. Studies (Zuo et al., 2020; Sawhney et al., 2021b) show that the Hawkes process can be used to model text sequences from social media and discourses. The hyperbolic Hawkes attention mechanism learns an excitation parameter ϵ corresponding to excitation induced by text p_j and a decay parameter α to learn the decay rate of this induced excitement. Formally, we use an Einstein midpoint (Ungar, 2005) to aggregate hidden states $\overline{h}$ via Hawkes process as,

$$\boldsymbol{u} = \texttt{HYPHEN}(\{p_i, t_i\}_{i=1}^T) = \sum_j \frac{\beta_j \gamma(\boldsymbol{q}_j)}{\sum_\tau \beta_\tau \gamma(\boldsymbol{q}_\tau)} \boldsymbol{q}_j \tag{8}$$

$$\boldsymbol{q_j} = \beta_j \odot \boldsymbol{h_j} \oplus \epsilon \odot \exp_o\left(\text{ReLU}(\log_o(\boldsymbol{h_j}))\right) \odot e^{-\alpha \Delta k} \tag{9}$$

where, $\gamma(\boldsymbol{q}_j) = \frac{1}{\sqrt{1 - ||\boldsymbol{q}_j||^2}}$ are the lorentz factors.

3 Applications and Tasks

Political Stance Prediction Parliamentary debates consist of responses from politicians over a motion. Following (Sawhney et al., 2020), we aim to classify the stance of a speaker as 'Aye'/'No' on a motion based on their historic speeches. We evaluate on the ParlVote dataset (Abercrombie and Batista-Navarro, 2020) comprising of 33,461 UK debate transcripts of 1,346 politicians.

Financial NLP We aim to predict future stock trends based on the historic texts about a stock. Following (Sawhney et al., 2021a) we regress the future volatility of a stock defined as $\lambda = ln(|\frac{p_i - p_{i-1}}{p_{i-1}}|)$, where p_i is the closing price. We evaluate on the S&P (Xu and Cohen, 2018) containing 88 stocks with 109,915 tweets and the China Stock Exchange (CSE) (Huang et al., 2018) containing 90,361 Chinese news articles for 85 stocks.

Suicide Ideation Following (Sawhney et al., 2021d), we aim to detect suicidal intent in a tweet given historic tweets from a user. We use the data from (Mishra et al., 2019) containing 32,558 user timelines and 2.3M texts.

Table 1: Performance comparison with baselines (mean of 40 runs). * indicates improvement over SOTA is significant ($p<0.01$) under Wilcoxon's signed rank test.

Model	PVote MCC ↑	SI MCC ↑	CSE MSE ↓	S&P MSE ↓
MLP(2018)	0.36	0.24	2.91	0.38
LSTM(1997)	0.52	0.28	2.88	0.34
HAN(2019)	0.50	0.29	2.85	0.31
H-LSTM(2020)	0.53	0.29	2.87	0.33
FAST(2021e)	0.51	0.30	2.86	0.32
HT-LSTM(2021a)	0.55	0.31	2.68	0.31
HYPHEN (Ours)	**0.63***	**0.44***	**2.68**	**0.29***

4 Results

4.1 Performance Comparison

We compare the performance of HYPHEN over financial, political, and healthcare tasks spanning English and Chinese languages in Table 1. We observe that HYPHEN generally outperforms most baseline methods by 10% on average. Overall, we note that methods that capture fine-grained timing irregularities in text sequences perform better (HYPHEN, FAST, HT-LSTM), validating our premise of using time-aware modeling. We postulate that HYPHEN's superior performance is due to, 1) learnable hyperbolic geometry and 2) time-aware hyperbolic Hawkes process. First, HYPHEN better encodes the varying hyperbolic properties of text sequences by learning a suitable data-driven curvature in contrast to other hyperbolic models (HT-LSTM), which constrain all sequences to a fixed hyperbolic space. Second, through hyperbolic time aware learning and Hawkes attention, HYPHEN better captures timing irregularities between the subsequent release of texts (Sawhney et al., 2021a). These observations collectively show the practical applicability and generalizability of HYPHEN for stream modeling.

4.2 Ablation Study

We contextualize the impact of various components of HYPHEN in Table 2. We note that augmenting RNN-based methods with attention leads to significant improvements ($p < 0.01$), as HYPHEN can better distinguish noise inducing text from relevant information (Sawhney et al., 2021e). Next, we observe significant ($p < 0.01$) improvements on using hyperbolic spaces to represent text streams, suggesting that the hyperbolic space better models the innate power-law dynamics and hierarchies in online text streams (Sala et al., 2018). Further, enriching the temporal attention with the Hawkes process leads to performance boosts, potentially because the Hawkes process better captures the excitation induced by influential texts. Finally, learning the underlying hyperbolic geometry benefits HYPHEN, allowing it to generalize to a variety of text streams with different hyperbolic properties.

Table 2: Ablation study over HYPHEN (mean of 40 runs). *,†indicate improvement over HYPHEN-constant curvature and Euclidean (EUC) counterparts are significant ($p < 0.01$) under Wilcoxon's signed rank test.

Ablation Components	PVote MCC↑	SI MCC↑	CSE MSE↓	S&P MSE↓
LSTM	0.52	0.28	2.88	0.34
EUC-Time LSTM+Attn	0.51	0.30	2.86	0.32
EUC-Time LSTM+Hwks	0.54	0.33	2.83	0.32
HYP-time LSTM + Attn	0.58†	0.31†	2.73†	0.31†
HYPHEN-constant curvature	0.61†	0.36†	2.72†	0.30†
HYPHEN (Ours)	**0.63***†	**0.44***†	**2.68***†	**0.29***†

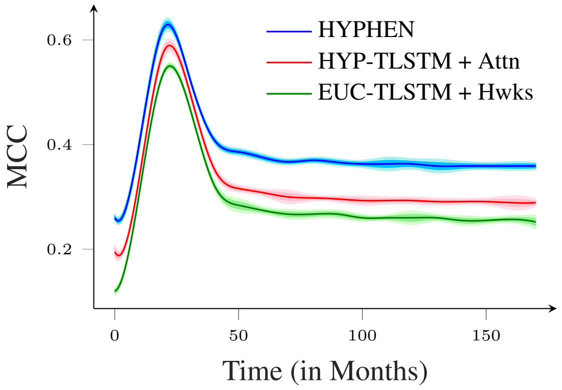


Figure 2: Sensitivity of HYPHEN to the lookback period T on political speaker state modeling.

4.3 Impact of Historical Context

We study the variation in HYPHEN's performance on political speaker state modeling corresponding to varying amounts of lookback periods T in Figure 2. First, without encoding the historic context, we observe that all models perform poorly. As we increase the lookback period, we note that Hawkes attention improves temporal attention, potentially because the Hawkes process decays the impact of very old texts enabling HYPHEN to focus on more recent debates which better reflects a speaker's temporal state. Further, with very large lookback periods, we observe a performance drop, likely because large amounts of context allow the inclusion of speeches from very old (stale) debates, which may not contribute significantly to the speaker's present state (Cullen et al., 2018). However, through hyperbolic Hawkes attention HYPHEN is able to filter out more crucial debates to an extent. In general, HYPHEN provides the best results with debates around ten months in the past (mid-sized lookbacks).

5 Conclusion

We explore the scale-free dynamics and timing irregularities of text streams. We propose HYPHEN which uses hyperbolic Hawkes attention and learns data-driven geometries to represent varying hyperbolic properties of streams. Through experiments on political, financial NLP, and healthcare tasks, we show the applicability of HYPHEN on 4 datasets.

Acknowledgements

We would like to thank the Financial Services Innovation Lab at Georgia Institute of Technology for their generous support.

6 Ethical Considerations

The sensitive nature of this work calls for careful deliberation of the risks and ethical challenges involved. While we only use publicly available user data, we emphasize the importance of preserving the privacy of the users involved (De Choudhury et al., 2016). We acknowledge that the predictive power of HYPHEN depends on the data, which is in tension with user privacy concerns. We carefully adopt the measures followed by Chancellor et al. (2016). Specifically, we operate within the acceptable privacy bounds (Chancellor et al., 2019) and considerations (Fiesler and Proferes, 2018) in order to avoid coercion and harmful interventions (Chancellor et al., 2019). We paraphrase and anonymize all samples in the suicide ideation detection detection dataset using the moderate disguise scheme (Bruckman, 2002; Fiesler and Proferes, 2018). We also perform automatic de-identification using named entity recognition to identify and mask personally identifiable information.

While one of our work's application is to aid in the early detection of suicidal users and early intervention, it is imperative that any interventions be well-thought, failing which may lead to counterhelpful outcomes, such as users moving to fringe platforms, which would make it harder to provide assistance (Kumar et al., 2015). Care should be taken so as not to create stigma, and interventions must be carefully planned by consulting relevant stakeholders such as clinicians, designers, and researchers (Chancellor et al., 2016), to maintain social media as a safe space for individuals looking to express themselves (Chancellor et al., 2019).

References

Gavin Abercrombie and Riza Batista-Navarro. 2018. 'aye' or 'no'? speech-level sentiment analysis of hansard uk parliamentary debate transcripts.

Gavin Abercrombie and Riza Batista-Navarro. 2020. ParlVote: A corpus for sentiment analysis of political debates. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5073–5078, Marseille, France. European Language Resources Association.

Inci M. Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K. Jain, and Jiayu Zhou. 2017. Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 65–74, New York, NY, USA. Association for Computing Machinery.

Gary Bécigneul and Octavian-Eugen Ganea. 2018. Riemannian adaptive optimization methods. *CoRR*, abs/1810.00760.

Amy Bruckman. 2002. Studying the amateur artist: A perspective on disguising data collected in human subjects research on the internet. *Ethics and Information Technology*, 4(3).

Stevie Chancellor, Michael L Birnbaum, Eric D Caine, Vincent MB Silenzio, and Munmun De Choudhury. 2019. A taxonomy of ethical tensions in inferring mental health states from social media. In *Proceedings of the conference on fairness, accountability, and transparency*.

Stevie Chancellor, Zhiyuan Lin, Erica L Goodman, Stephanie Zerwas, and Munmun De Choudhury. 2016. Quantifying and predicting mental illness severity in online pro-eating disorder communities. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*.

Ailbhe Cullen, Andrew Hines, and Naomi Harte. 2018. Perception and prediction of speaker appeal – a single speaker study. *Computer Speech & Language*, 52:23 – 40.

Munmun De Choudhury, Emre Kiciman, Mark Dredze, Glen Coppersmith, and Mrinal Kumar. 2016. Discovering shifts to suicidal ideation from mental health content in social media. In *Proceedings of the 2016 CHI conference on human factors in computing systems*.

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

Hans J Eysenck. 1968. *The psychology of politics*, volume 2. Transaction publishers.

Casey Fiesler and Nicholas Proferes. 2018. “participant” perceptions of twitter research ethics. *Social Media+ Society*, 4(1):2056305118763366.

Thierry Foucault, Johan Hombert, and Ioanid Roşu. 2016. News trading and speed. *The Journal of Finance*, 71(1):335–382.

Xavier Gabaix. 2016. Power laws in economics: An introduction. *Journal of Economic Perspectives*, 30(1):185–206.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. In *NeurIPS*, pages 5350–5360.

Albert Gu, Frederic Sala, Beliz Gunel, and Christopher Ré. 2019. Learning mixed-curvature representations in product spaces. In *International Conference on Learning Representations*.

Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, and Nando de Freitas. 2019. Hyperbolic attention networks. In *International Conference on Learning Representations*.

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

Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. 2018. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM ’18, page 261–269, New York, NY, USA. Association for Computing Machinery.

Jieyun Huang, Yunjia Zhang, Jialai Zhang, and Xi Zhang. 2018. A tensor-based sub-mode coordinate algorithm for stock prediction.

Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. 2020. Hyperbolic image embeddings. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mrinal Kumar, Mark Dredze, Glen Coppersmith, and Munmun De Choudhury. 2015. Detecting changes in suicide content manifested in social media following celebrity suicides. In *Proceedings of the 26th ACM conference on Hypertext & Social Media*, pages 85–94.

Tara Law. 2019. The most important presidential debates in american history, according to historians. `https://time.com/5607429/most-important-debates/`. Accessed: 2021-09-15.

Federico López and Michael Strube. 2020. A fully hyperbolic neural model for hierarchical multi-class classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 460–475, Online. Association for Computational Linguistics.

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

Hongyuan Mei and Jason Eisner. 2017. The neural hawkes process: A neurally self-modulating multivariate point process.

Rohan Mishra, Pradyumn Prakhar Sinha, Ramit Sawhney, Debanjan Mahata, Puneet Mathur, and Rajiv Ratn Shah. 2019. SNAP-BATNET: Cascading author profiling and social network graphs for suicide ideation detection on social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 147–156, Minneapolis, Minnesota. Association for Computational Linguistics.

Nuno Oliveira, Paulo Cortez, and Nelson Areal. 2017. The impact of microblogging data for stock market prediction: Using twitter to predict returns, volatility, trading volume and survey sentiment indices. *Expert Systems with Applications*, 73:125–144.

Frederic Sala, Christopher De Sa, Albert Gu, and Christopher Ré. 2018. Representation tradeoffs for hyperbolic embeddings. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4457–4466. PMLR.

Ramit Sawhney, Shivam Agarwal, Megh Thakkar, Arnav Wadhwa, and Rajiv Ratn Shah. 2021a. *Hyperbolic Online Time Stream Modeling*, page 1682–1686. Association for Computing Machinery, New York, NY, USA.

Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, Tyler Derr, and Rajiv Ratn Shah. 2021b. Stock selection via spatiotemporal hypergraph attention network: A learning to rank approach. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(1):497–504.

Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, and Rajiv Shah. 2021c. Tec: A time evolving contextual graph model for speaker state analysis in political debates. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3552–3558. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Ramit Sawhney, Harshit Joshi, Rajiv Ratn Shah, and Lucie Flek. 2021d. Suicide ideation detection via social and temporal user representations using hyperbolic learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2176–2190, Online. Association for Computational Linguistics.

Ramit Sawhney, Arnav Wadhwa, Shivam Agarwal, and Rajiv Ratn Shah. 2020. GPolS: A contextual graph-based language model for analyzing parliamentary debates and political cohesion. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4847–4859, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ramit Sawhney, Arnav Wadhwa, Shivam Agarwal, and Rajiv Ratn Shah. 2021e. FAST: Financial news and tweet based time aware network for stock trading. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2164–2175, Online. Association for Computational Linguistics.

Martin Scholtus, Dick van Dijk, and Bart Frijns. 2014. Speed, algorithmic trading, and market quality around macroeconomic news announcements. *Journal of Banking and Finance*, 38:89 – 105.

Ryohei Shimizu, YUSUKE Mukuta, and Tatsuya Harada. 2021. Hyperbolic neural networks++. In *International Conference on Learning Representations*.

Alexandru Tifrea, Gary Becigneul, and Octavian-Eugen Ganea. 2019. Poincare glove: Hyperbolic word embeddings. In *International Conference on Learning Representations*.

Abraham A Ungar. 2005. *Analytic hyperbolic geometry: Mathematical foundations and applications*. World Scientific.

Teun A Van Dijk. 2002. Political discourse and political cognition. *Politics as text and talk: Analytic approaches to political discourse*, 203:203–237.

Teun Adrianus Van Dijk. 1977. *Text and context: Explorations in the semantics and pragmatics of discourse*. Longman London.

Yumo Xu and Shay B. Cohen. 2018. Stock movement prediction from tweets and historical prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1970–1979, Melbourne, Australia. Association for Computational Linguistics.

Xiaojun Zhao, Pengjian Shang, and Yulei Pang. 2010. Power law and stretched exponential effects of extreme events in chinese stock markets. *Fluctuation and Noise Letters*.

Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. 2020. Transformer hawkes process. In *International Conference on Machine Learning*, pages 11692–11702. PMLR.

A Experimental Setup

A.1 Datasets

- **US S&P (Xu and Cohen, 2018):** US S&P stocks are categorized into 9 industries: basic materials, consumer goods, healthcare, services, utilities, conglomerates, financial, industrial goods and technology. US S&P dataset contains text data and historical prices of 88 stocks which includes all 8 stocks in conglomerates and the top 10 stocks by market capitalization in each of the other industries. The text data comprises tweets from 01/01/2014 to 01/01/2016. Following (Xu and Cohen, 2018) we split the US S&P temporally based on date ranges from 01/01/2014 to 01/08/2015 for training, 01/08/2015 to 01/10/2015 for validation, and 01/10/2015 to 01/01/2016 for test.

- **China and Hong Kong (CSE) (Huang et al., 2018):** China and Hong Kong (CSE) dataset consists of news headlines of 85 top-traded stocks listed on the Shanghai, Shenzhen, and Hong Kong Stock Exchange from January 2015 to December 2015. The qualitative data comprises of 90,361 Chinese financial news headlines. We split the China & HK dataset temporally based on date ranges from 01/01/2015 to 31/08/2015 for training, 01/09/2015 to 30/09/2015 for validation, and 01/10/2015 to 01/01/2016 for testing all models.

- **ParlVote (Abercrombie and Batista-Navarro, 2020):** Following (Sawhney et al., 2020) we evaluate political stance detection on the ParlVote dataset. This record consists of debate transcripts from the UK House of Commons obtained under an open Parliament license. Following (Abercrombie and Batista-Navarro, 2020) we remove non-speech elements from the transcripts and the original casing is preserved. ParlVote consists of 33,461 transcripts from May 7$^{\text{th}}$ 1997 to November 5$^{\text{th}}$ 2019. The average number of tokens in a ParlVote speech is 760.2 ± 901.3. Based on a speaker's vote to their speech, transcripts are labeled as 'Aye' and 'No' representing positive and negative stance respectively. The dataset is fairly balanced, consisting of 53.57% ‘Aye’ and 46.43% ‘No’ labels. We split the dataset temporally to obtain 70%, 15% and 15% of the data for training, validation and testing respectively.

- **Suicide Ideation.** (**Sawhney et al., 2021d**): The Suicide ideation dataset is built upon the existing Twitter tweets database of (Mishra et al., 2019). The dataset consists of tweets of 32,558 unique users, spanning over ten years of historical tweets from 2009 to 2019. Out of all the tweets, 34,306 tweets were identified as having potential suicide ideation words. These tweets were then manually annotated by two psychologists under the supervision of a head psychologist and 3984 tweets were actually identified as having suicidal tendencies. The same preprocessing techniques were employed on the dataset as done by Sawhney et al. (2021d).

A.2 Evaluation Metrics

Matthews correlation coefficient: The Matthews correlation coefficient (MCC) produces a high score only if the prediction obtained good results in all of the four confusion matrix categories (true positives, false negatives, true negatives, and false positives), proportionally both to the size of positive elements and the size of negative elements in the dataset. We use MCC to evaluate on suicide ideation detection and political speech classification.

Mean squared error: To evaluate the volatility regression performance, we adopt the Mean Squared Error (MSE) to compute the error between actual and the predicted volatility values.

A.3 Baseline Models

We compare HYPHEN with the following baselines:

- **MLP**: A Bag of Words model that uses unigram textual features as input along with the TF-IDF vectors which are fed into a multi-layer perceptron (Abercrombie and Batista-Navarro, 2020).

- **LSTM** : An RNN architecture capable of learning long term sequential dependencies (Hochreiter and Schmidhuber, 1997).

- **HAN**: Transformer model with hyperbolic activations and attention which utilises hyperbolic geometry for both computation and aggregation of attention weights (Gulcehre et al., 2019).

- **H-LSTM**: A RNN based model for sequential data with an attention mechanism operating in the hyperbolic space (López and Strube, 2020).

- **FAST**: A time-aware LSTM network capable of modeling the fine grained temporal irregularities in textual data (Sawhney et al., 2021e).

- **HT-LSTM**: Hierarchical Time-aware hyperbolic LSTM network leverages the hyperbolic space for encoding scale-free nature of a text stream (Sawhney et al., 2021a).

A.4 Training Setup

We have performed all our experiments on Tesla GPU. We performed a grid search for all our models and selected the best values based on the validation MCC/MSE. We followed the same preprocessing techniques as suggested by the dataset authors. We explored the lookback window length $T \in [2, 20]$ and the hidden state dimensions in $\in (64, 128, 256)$. We grid searched our learning rates in $\in (1e-5, 5e-4, 1e-3)$. We used Riemannian Adam (Bécigneul and Ganea, 2018) as our optimizer.

A Risk-Averse Mechanism for Suicidality Assessment on Social Media

Ramit Sawhney[1]*, **Atula Tejaswi Neerkaje**[2]*, **Manas Gaur**[1]
[1]AI Institute, University of South Carolina, SC, USA
mgaur@email.sc.edu
[2]Manipal Institute of Technology, Manipal, India
atula.neerkaje@learner.manipal.edu

Abstract

Recent studies have shown that social media has increasingly become a platform for users to express suicidal thoughts outside traditional clinical settings. With advances in Natural Language Processing strategies, it is now possible to design automated systems to assess suicide risk. However, such systems may generate uncertain predictions, leading to severe consequences. We hence reformulate suicide risk assessment as a selective prioritized prediction problem over the Columbia Suicide Severity Risk Scale (C-SSRS). We propose SASI, a risk-averse and self-aware transformer-based hierarchical attention classifier, augmented to refrain from making uncertain predictions. We show that SASI is able to refrain from 83% of incorrect predictions on real-world Reddit data. Furthermore, we discuss the qualitative, practical, and ethical aspects of SASI for suicide risk assessment as a human-in-the-loop framework.

1 Introduction

Suicide is a global phenomenon responsible for 1.3% of deaths worldwide (WHO, 2019). While it is the leading cause of death among 14-35 year olds in the US (Hedegaard et al., 2021), suicide rates have increased by 13% in Japan between July to September 2020 (Tanaka and Okamoto, 2021). It hence becomes critical to extend clinical and psychiatric care, which relies heavily on identifying those at risk. While 80% of patients do not undergo clinical treatment, 60% of those who succumbed to suicide denied having suicidal thoughts to mental health experts (McHugh et al., 2019). However, studies show eight out of ten people shared suicidal thoughts on social media (Golden et al., 2009).

The advent of Natural Language Processing (NLP) shows promise for suicide risk assessment based on online user behavior (Ji et al., 2021b; Choudhury et al., 2016), with automatic risk assessment algorithms outperforming traditional clinical methods (Coppersmith et al., 2018; Linthicum et al., 2019). Numerous deep learning methods already exist, which include leveraging suicide-related word-embeddings (Cao et al., 2019), social graphs (Mishra et al., 2019; Sinha et al., 2019; Cao et al., 2022; Sawhney et al., 2021b) and historical context (Matero et al., 2019; Gaur et al., 2019).

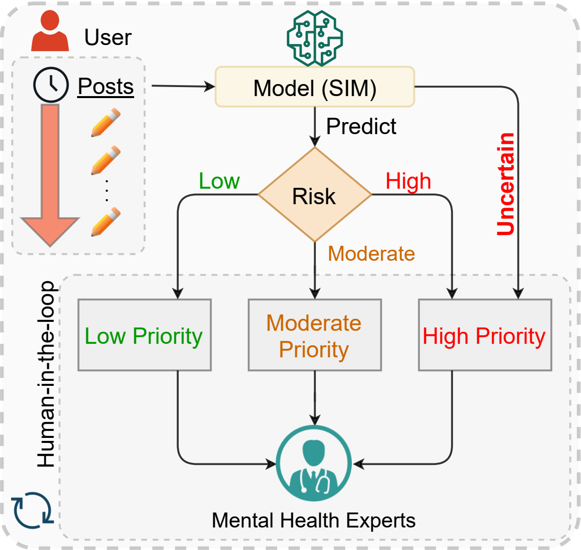


Figure 1: End-to-end pipeline for suicide risk assessment. When SASI assesses the posts, it returns the predicted risk level along with a certainty score. With a human-in-the-loop framework, these predictions can be sorted into various risk levels. SASI assigns high priority to uncertain predictions, for an immediate review by mental health experts.

However, mental health is a safety-critical realm, where technological failure could lead to severe harm to users on social media (Sittig and Singh, 2015). One such case was covered by Register (2020), wherein a medical bot suggested a mock patient kill themselves, demonstrating that unintended harmful behavior can emerge from AI systems (Amodei et al., 2016; Chandler et al., 2020).

[*] Authors contributed equally

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 628 - 635
May 22-27, 2022 ©2022 Association for Computational Linguistics

Despite the significant power of traditional NLP methods, such models are inherently designed to make a prediction even when not confident. This poses a challenge when working with critical tasks like suicide risk assessment, for which it may be hard to make a prediction due to various reasons such as task hardness or contained ambiguity. Such a system may associate a lower risk level to a user who needs urgent help. A resulting delayed response from mental health experts may lead to adverse consequences. We hence need systems that assign high priority to uncertain predictions, for immediate review and response.

Contributions: We reformulate suicide risk assessment as a prioritized prediction task which factors in uncertainty, and propose **SASI**: A Risk-Averse Mechanism for **S**uicidality **A**ssessment on **S**ocial Med**I**a. SASI is risk-averse in the sense that it is self-aware, as it incorporates a selection function to measure uncertainty. Based on a set threshold value, SASI refrains from making a prediction when it is uncertain. We show that SASI can act as a tool to efficiently prioritize users who need immediate attention. Through a human-in-the-loop framework that involves a domain expert, SASI assigns high priority to uncertain predictions to avoid critical failure (Figure 1). We demonstrate the effectiveness of SASI using a real-world gold standard Reddit dataset. Through a series of experiments, we show SASI refrains from making 83% of incorrect predictions. We further demonstrate its effectiveness through a qualitative study and discuss the ethical implications.

2 Methodology

2.1 Columbia Suicide Severity Risk Scale

The Columbia Suicide Severity Rating Scale (C-SSRS) is an authoritative questionnaire employed by psychiatrists to measure suicide risk severity (Posner et al., 2011). There are 3 items in the scale: Suicide Ideation, Suicide Behavior, and Suicide Attempt. Each C-SSRS severity class is composed of a conceptually organized set of questions that characterize the respective category. Responses to the questions across the C-SSRS classes eventually determine the risk of suicidality of an individual (Interian et al., 2018; McCall et al., 2021). One of the challenges researchers face when it comes to dealing with social media content is the disparity in the level of emotions expressed (Gaur et al., 2019). Since the C-SSRS was originally designed for use in clinical settings, adapting the same metric to a social media platform would require changes to address the varying nature of emotions expressed. For instance, while in a clinical setting, it is typically suicidal candidates that see a clinician; on social media, non-suicidal users may participate to offer support to others deemed suicidal (Gaur et al., 2021). To address these factors, two additional classes were defined (Gaur et al., 2019) to the existing C-SSRS scale with three classes: Suicide Indicator and Supportive (Negative class).

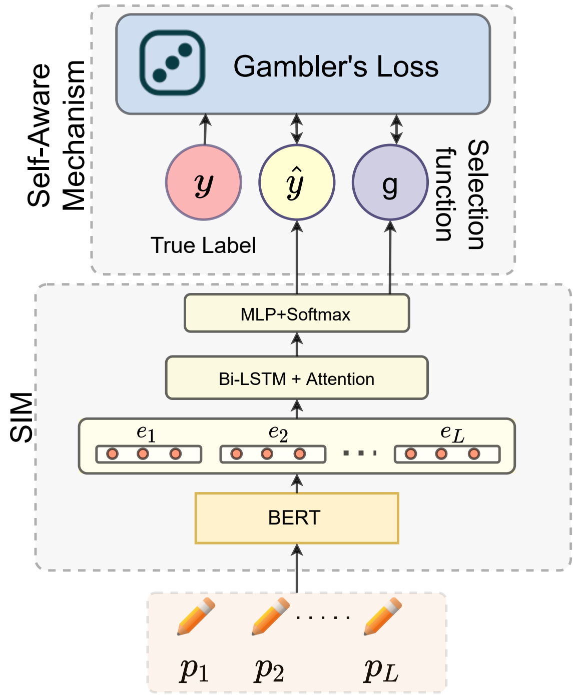


Figure 2: An overview of SASI: SASI incorporates a risk-averse, self-aware mechanism to any given suicide ideation model (SIM) by training using Gambler's Loss. It refrains from predicting when uncertain.

2.2 Problem Formulation

Following existing work (Gaur et al., 2019; Sawhney et al., 2021a), we formulate the problem as a classification task to predict the suicidal risk of the user $u_i \in \{u_1, u_2, \cdots, u_N\}$, whose posts $P_i = \{p^i_1, p^i_2, \cdots, p^i_T\}$ are authored over time in a chronological order, with the latest post being p^i_T. We denote the label set $\mathbf{Y}$ = {Support (SU), Indicator (IN), Ideation (ID), Behaviour (BR), Attempt (AT)} in increasing order of severity risk, defined based on the C-SSRS. For a given Suicide Ideation Model, our goal is to expand the cardinality of the label space to $|\mathbf{Y}| + 1$ so as to enable an option to refrain when the model is uncertain.

2.3 Suicide Ideation Model (SIM)

Each post made by a user could provide detailed context of suicidal thought manifestation over time (Oliffe et al., 2012). To capture this property, we draw inspiration from existing state-of-the-art (SOTA) models (Gaur et al., 2019; Matero et al., 2019; Sawhney et al., 2021a; Ji et al., 2021a) which use LSTM based backbones. To encode each post p_k^i, we use the 768-dimensional representation of the `[CLS]` token obtained from BERT (Devlin et al., 2019) as e_k^i=BERT(p_k^i). As shown in Figure 2, we then pass each post embedding sequentially through a bi-directional LSTM, given as h_k^i = Bi-LSTM(e_k^i). We thus obtain the sequence of hidden states, $\boldsymbol{x} = [h_1^i, h_2^i, \cdots, h_T^i]$, where $h_k^i \in \mathbb{R}^H$, and H is the hidden dimension. To filter out relevant signals from the potentially vast user history (Shing et al., 2020), we pass the hidden state sequence through an attention layer. The final layer is a multilayer perceptron (MLP) to obtain the prediction vector $\hat{\boldsymbol{y}}$, given as:

$$\begin{aligned}&\hat{\boldsymbol{y}} = f(\boldsymbol{x}), \quad \text{where}\\ &f(\boldsymbol{x}) = \text{Softmax}(\text{MLP}(\text{Attention}(\boldsymbol{x})))\end{aligned} \tag{1}$$

2.4 Self-Aware Mechanism

To make the model self-aware, we transform the model such that it makes a prediction only when certain (Liu et al., 2019). As shown in Figure 2, the model $f : \mathbb{R}^{T\times H} \to \mathbf{Y}$ is augmented with a selection function $g : \mathbb{R}^{T\times H} \to (0, 1)$, which is an extra logit. The augmented model is described as a piece-wise function, given by:

$$(f, g)(\boldsymbol{x}) := \begin{cases}\text{Refrain}, & \text{if } g \geq \tau\\ \text{argmax}(\hat{\boldsymbol{y}}), & \text{otherwise}\end{cases} \tag{2}$$

Where the threshold $\tau \in (0, 1)$, argmax$(\hat{\boldsymbol{y}}) \in \mathbf{Y}$. Let $p = (f, g)(\boldsymbol{x})$, where $p \in \mathbf{Y} \cup \{\text{Refrain}\}$ denote the final prediction by the model for a user u_i. Human moderators can then define the level of granularity of these predictions, and sort them into priority levels as desired. As an example, moderators may choose to have only three levels of priority, where the user is high priority if $p \in$ {AT, BR, Refrain}, moderate if $p \in$ {ID, IN} and low if $p \in$ {SU}. With the addition of the Refrain option, uncertain predictions will have highest priority, alleviating the possibility of high-risk users being neglected.

It is essential to note that the confidence threshold τ is not utilized during training, rather as a threshold variable to calibrate data coverage (cov) during evaluation. The cov fraction of total samples is what SASI predicts on, leaving out $(1 - cov)$ samples for which SASI is most uncertain. Specifically, we can choose some value τ such that there will be $(1 - cov)$ samples for which $g \geq \tau$. The idea behind this approach is to trade-off $(1 - cov)$ samples for immediate review by mental health experts in exchange for higher model performance on the cov samples about which it is confident.

2.5 Network Optimization

In any m-class classification problem, if the model assigns a high probability score to the wrong class, then learning becomes difficult due to vanishing gradients (Ziyin et al., 2020). To account for the additional refrain option in the augmented label space, we train SASI using Gambler's Loss (Liu et al., 2019). Gambler's loss allows the gradients to propagate through g instead, by abstaining from assigning weights to any of the m classes. Thus, the model learns a distribution of noisy/uncertain data points characterized by the selection function g. The loss function is given as:

$$\mathcal{L} = -\sum_{j}^{|\mathbf{Y}|} y_j \cdot \log(\hat{y}_j \cdot r + g) \tag{3}$$

where y_j is the true label, and the reward r is a hyperparameter. A higher value of r discourages restraint. Since the loss function directly learns g, it does not depend on the coverage (Liu et al., 2019), and can be manually set to any value during evaluation.

3 Experimental Setup

3.1 Dataset

We use the dataset released by Gaur et al. (2019), which contains Reddit posts of 500 users filtered from an initial set of 270,000 users across several mental health and suicide-related subreddits, such as r/StopSelfHarm (SSH), r/selfharm (SLF), r/bipolar (BPL), r/BipolarReddit (BPR), r/BipolarSOs, r/opiates (OPT), r/Anxiety (ANX), r/addiction (ADD), r/BPD, r/SuicideWatch (SW), r/schizophrenia (SCZ), r/autism (AUT), r/depression (DPR), r/cripplingalcoholism (CRP), and r/aspergers (ASP). The posts were annotated by practicing psychiatrists into five increasing risk levels based on the Columbia Suicide Severity Risk Scale (Posner et al., 2011), leading to an acceptable average pairwise agreement of 0.79 and a groupwise agreement of 0.73. The class distribution of each category with increasing risk level is: Supportive (20%), Indicator (20%), Ideation (34%), Behaviour (15%), Attempt (9%). On average, the number of posts made by a user is 18.25±27.45 with a maximum of 292 posts. The average number of tokens in each post is 73.4±97.7.

3.2 Evaluation Metrics

We first describe the evaluation metrics that measure how well the model performs on the *cov* samples. Following Gaur et al. (2019), we use graded variants of F1 score, Precision, and Recall, where we alter the formulation of False Negatives (FN) and False Positives (FP). FN is modified as the ratio of the number of times predicted severity of suicide risk level (k^p) is less than the actual risk level (k^a) over N number of samples. FP is the ratio of the number of times the predicted risk (k^p) is greater than the actual risk (k^a), given as:

$$\begin{aligned} FN &= \frac{\sum_{i=1}^{N} I(k_i^a > k_i^p)}{N} \\ FP &= \frac{\sum_{i=1}^{N} I(k_i^p > k_i^a)}{N} \end{aligned} \tag{4}$$

Let P_{T} denote the total number of test samples, $P_{\text{corr+refrain}}$ the sum of samples that have either been correctly predicted or have been refrained, P_{refrain} the total number of refrained samples, and P_{in} the number of incorrect predictions among the refrained samples. We additionally introduce two metrics, *Robustness* and *Fail-Safe Rejects*, as:

$$\begin{aligned} \text{Robustness} &= \frac{P_{\text{corr+refrain}}}{P_{\text{T}}} \\ \text{Fail-Safe Rejects} &= \frac{P_{\text{in}}}{P_{\text{refrain}}} \end{aligned} \tag{5}$$

Robustness captures the fraction of samples which are correctly classified or instead sent for immediate review. *Fail-Safe Rejects* captures the fraction of refrained samples which were indeed erroneous. A higher Fail-Safe Rejects score hence implies that human moderators will be subjected to a lesser amounts of redundant work.

4 Results

4.1 Performance Comparison

We compare the performance of SASI with various state-of-the-art baselines in Table 1. Sequential models like Suicide Detection Model (SDM) (Cao et al., 2019) and ContextBERT (Matero et al., 2019) generally outperform ContextualCNN (Gaur et al., 2019), which uses a bag-of-posts approach. SISMO (Sawhney et al., 2021a) shows further improvements by modeling the ordinal nature of risk labels. SASI significantly outperforms ($p < 0.005$) these methods for various values of coverage (*cov*), demonstrating its ability to avoid committing to erroneous predictions by characterizing its confidence (Liu et al., 2019).

Model	**Gr. Prec.**	**Gr. Recall**	**FScore**	**Robustness**	**Fail-Safe Rejects**
Contextual CNN	0.65	0.52	0.59	-	-
SDM	0.61	0.54	0.57	-	-
ContextBERT	0.63	0.57	0.60	-	-
SISMO	0.66	0.61	0.64	-	-
SASI (Cov 100%)	0.67*	0.62	0.66*	0.48	-
SASI (Cov 85%)	*0.69**	*0.65**	*0.67**	*0.61*	**0.83**
SASI (Cov 50%)	**0.71***	**0.69***	**0.70***	**0.73**	*0.65*

Table 1: We report the median of results over 10 random seeds. * indicates the result is statistically significant with respect to SISMO ($p < 0.005$) under Wilcoxon's signed-rank test. **Bold** denotes best performance while *Italics* denotes second best.

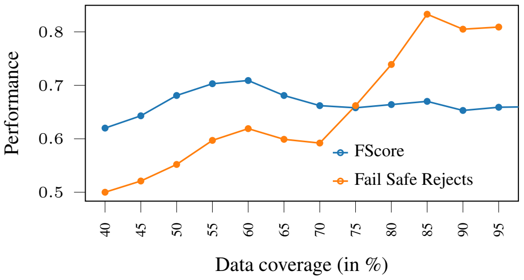


Figure 3: Changes in performance metrics with increasing coverage, averaged over 10 random seeds.

4.2 Coverage and Performance Trade-off

We further evaluate SASI for various values of target coverage (*cov*) by calibrating the threshold τ. As shown in Figure 3, lower coverage leads to an increase in Graded Recall, Precision, and FScore (Table 1), as the model only keeps *cov* predictions which it is highly certain about. However, we observe a decrease in Fail-Safe Rejects due to an increasingly cautious approach employed by the model, which implies an increased fraction of originally correct predictions that need to be manually reviewed. We hence observe a trade-off, wherein we must seek to achieve competitive performance on the *cov* samples, while at the same time not overburden moderators with the $(1 - cov)$ samples. For lower coverage values (say 50%), human modera-

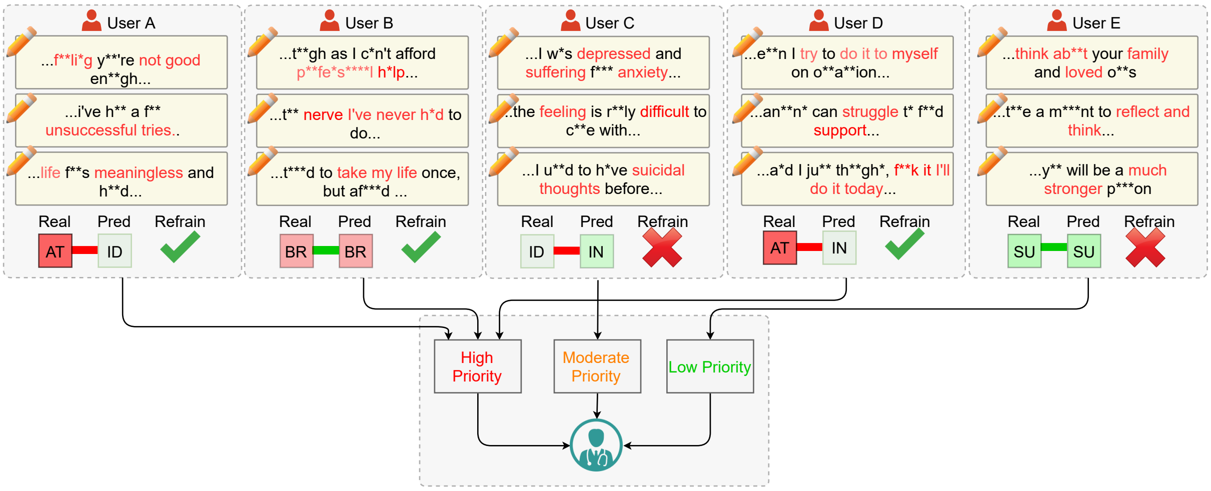

User A
...f**li*g y**'re not good en**gh...
...i've h** a f** unsuccessful tries..
...life f**s meaningless and h**d...
Real
Pred
Refrain
AT
ID
✓
User B
...t**gh as I c*n't afford p**fe*s****l h*lp...
...t** nerve I've never h*d to do...
...t***d to take my life once, but af***d ...
Real
Pred
Refrain
BR
BR
✓
User C
...I w*s depressed and suffering f*** anxiety...
...the feeling is r**ly difficult to c**e with...
...I u**d to h*ve suicidal thoughts before...
Real
Pred
Refrain
ID
IN
✗
User D
...e**n I try to do it to myself on o**a**ion...
...an**n* can struggle t* f**d support...
...a*d I ju** th**gh*, f**k it I'll do it today...
Real
Pred
Refrain
AT
IN
✓
User E
...think ab**t your family and loved o**s
...t**e a m***nt to reflect and think...
...y** will be a much stronger p***on
Real
Pred
Refrain
SU
SU
✗
High Priority
Moderate Priority
Low Priority

Figure 4: We show SASI can be used for efficient prioritization of users during suicide risk assessment. For each user, we show the real labels next to predicted labels, while also indicating whether SASI refrained from making that prediction. We further demonstrate how SASI sorts the users into priority levels. All examples in this paper have been paraphrased as per the moderate disguise scheme (Bruckman, 2002) to protect user privacy.

tors may be overburdened by having to review a lot of redundant samples. On the other hand, we note that SASI (85%) provides more utility, as it statistically outperforms SOTA models like SISMO, while maintaining a fail-safe rejection score of 83% and a competitive robustness score of 61%.

4.3 Qualitative Analysis

The essence of SASI lies behind its ability to refrain from making misleading predictions over high-risk samples. We study five users with snippets of their posts, as shown in Figure 4. We observe the model makes erroneous predictions on high-risk users A and D. However, SASI refrains from committing to these predictions, assigning these users a high priority for immediate review and response. SASI chooses to refrain despite predicting the risk level of user B correctly, possibly because it employs a cautious approach due to phrases such as '*take my life*' scattered in the user's timeline. This user, who is already of relatively high risk, is hence assigned a high priority. User E shows a very low sign of risk, which is confidently captured by SASI without needing to refrain. User C is an erroneous case wherein SASI is confident, yet makes a wrong prediction. However, the user is not high risk and gets assigned to the same priority level as the true risk label. While this example is not a cause for concern, certain situations may arise where SASI also confidently assigns a low-risk score to a high-risk user, opening avenues for future work that involves integrating and reformulating ordinal regression over the principles of Gambler's loss.

5 Conclusion

With a motivation to provide a robust solution to fine-grained suicide risk assessment on social media, we present SASI, a framework that integrates the concept of selective prioritization to existing deep learning based risk-assessment techniques. SASI is self-aware, wherein it refrains from making a prediction when uncertain, and instead assigns high priority to such data samples for immediate review by mental health experts. We demonstrated the effectiveness of SASI through quantitative evaluations on real-world data, wherein SASI avoided high-risk situations by refraining from making 83% of incorrect predictions. Through a qualitative analysis, we described how SASI can be used as a part of a human-in-the-loop framework, facilitating efficient responses from mental health experts.

Acknowledgements

We thank Prof. Amit Sheth for reviewing the paper and providing valuable feedback and support. We would also like to thank the anonymous reviewers for their insightful suggestions on various aspects of this work.

Ethical Considerations

We work within the scope of acceptable privacy practices suggested by Chancellor et al. (2019) and considerations presented by Fiesler and Proferes (2018) to avoid coercion and intrusive treatment. The primary source of the dataset used in this study is Reddit. Although Reddit is intended for anonymous posting, we take further precautions by performing automatic de-identification of the dataset using named entity recognition (Zirikly et al., 2019). All examples used in this paper are further been anonymized, obfuscated, and paraphrased for user privacy (Benton et al., 2017) and to prevent misuse as per the moderate disguise scheme suggested by Bruckman (2002). Taking inspiration from Benton et al. (2017), we also keep the annotation of user data separate from raw user data on protected servers linked only through anonymous IDs. Our work focuses on building an assistive tool for screening suicidal users and providing judgments purely based on observational capacity. We acknowledge that it is almost impossible to prevent abuse of released technology even when developed with good intentions (Hovy and Spruit, 2016). Hence, we ensure that this analysis is shared only selectively to avoid misuse such as Samaritan's Radar (Hsin et al., 2016).

We further acknowledge that the studied data may be susceptible to demographic, expert annotator, and medium-specific biases (Hovy and Spruit, 2016). While the essence of our work is to aid in the early detection of at-risk users and early intervention, any interventions must be well-thought, failing which may lead to counter-helpful outcomes, such as users moving to fringe platforms, making it harder to provide assistance (Kumar et al., 2015). Care should be taken to not to create stigma, and interventions must hence be carefully planned by consulting relevant stakeholders, such as clinicians, designers, and researchers (Chancellor et al., 2016), to maintain social media as a safe space for individuals looking to express themselves (Chancellor et al., 2019). It is also essential that clinicians and human moderators are not overburdened (Chancellor et al., 2019). For instance, "Alarm fatigue" is when alarms are so excessive, many of which are false positives, that healthcare providers become desensitized from alarms (Drew et al., 2014).

We also agree that suicidality is subjective (Keilp et al., 2012), wherein the interpretation may vary across individuals on social media (Puschman, 2017). We do not make any diagnostic claims, rather help prioritize the users that should be evaluated by the medical professionals first, as part of a distributed human-in-the-loop framework (de Andrade et al., 2018).

References

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in ai safety. *ArXiv preprint*, abs/1606.06565.

Adrian Benton, Glen Coppersmith, and Mark Dredze. 2017. Ethical research protocols for social media health research. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 94–102, Valencia, Spain. Association for Computational Linguistics.

Amy Bruckman. 2002. Studying the amateur artist: A perspective on disguising data collected inhuman subjects research on the internet. *Ethics and Inf. Technol.*, 4(3):217–231.

Lei Cao, Huijun Zhang, and Ling Feng. 2022. Building and using personal knowledge graph to improve suicidal ideation detection on social media. *IEEE Transactions on Multimedia*, 24:87–102.

Lei Cao, Huijun Zhang, Ling Feng, Zihan Wei, Xin Wang, Ningyun Li, and Xiaohao He. 2019. Latent suicide risk detection on microblog via suicide-oriented word embeddings and layered attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1718–1728, Hong Kong, China. Association for Computational Linguistics.

Stevie Chancellor, Michael L. Birnbaum, Eric D. Caine, Vincent M. B. Silenzio, and Munmun De Choudhury. 2019. A taxonomy of ethical tensions in inferring mental health states from social media. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, page 79–88, New York, NY, USA. Association for Computing Machinery.

Stevie Chancellor, Zhiyuan Lin, Erica L. Goodman, Stephanie Zerwas, and Munmun De Choudhury. 2016. Quantifying and predicting mental illness severity in online pro-eating disorder communities. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*, CSCW '16, page 1171–1184, New York, NY, USA. Association for Computing Machinery.

Chelsea Chandler, Peter W Foltz, and Brita Elvevåg. 2020. Using machine learning in psychiatry: the need to establish a framework that nurtures trustworthiness. *Schizophrenia bulletin*, 46(1):11–14.

Munmun De Choudhury, Emre Kiciman, Mark Dredze, Glen Coppersmith, and Mrinal Kumar. 2016. Discovering shifts to suicidal ideation from mental health content in social media. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, San Jose, CA, USA, May 7-12, 2016*, pages 2098–2110. ACM.

Glen Coppersmith, Ryan Leary, Patrick Crutchley, and Alex Fine. 2018. Natural language processing of social media as screening for suicide risk. *Biomedical informatics insights*, 10:1178222618792860.

Norberto Nuno Gomes de Andrade, Dave Pawson, Dan Muriello, Lizzy Donahue, and Jennifer Guadagno. 2018. Ethics and artificial intelligence: suicide prevention on facebook. *Philosophy & Technology*, 31(4):669–684.

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

Barbara J Drew, Patricia Harris, Jessica K Zègre-Hemsey, Tina Mammone, Daniel Schindler, Rebeca Salas-Boni, Yong Bai, Adelita Tinoco, Quan Ding, and Xiao Hu. 2014. Insights into the problem of alarm fatigue with physiologic monitor devices: a comprehensive observational study of consecutive intensive care unit patients. *PloS one*, 9(10):e110274.

Casey Fiesler and Nicholas Proferes. 2018. "participant" perceptions of twitter research ethics. *Social Media + Society*, 4(1):2056305118763366.

Manas Gaur, Amanuel Alambo, Joy Prakash Sain, Ugur Kursuncu, Krishnaprasad Thirunarayan, Ramakanth Kavuluru, Amit P. Sheth, Randy S. Welton, and Jyotishman Pathak. 2019. Knowledge-aware assessment of severity of suicide risk for early intervention. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 514–525. ACM.

Manas Gaur, Vamsi Aribandi, Amanuel Alambo, Ugur Kursuncu, Krishnaprasad Thirunarayan, Jonathan Beich, Jyotishman Pathak, and Amit Sheth. 2021. Characterization of time-variant and time-invariant assessment of suicidality on reddit using c-ssrs. *PloS one*, 16(5):e0250448.

Robert N Golden, Carla Weiland, and Fred Peterson. 2009. *The truth about illness and disease*. Infobase Publishing.

Holly Hedegaard, Sally C Curtin, and Margaret Warner. 2021. Suicide mortality in the united states, 1999-2019. *NCHS data brief*, (398):1–8.

Dirk Hovy and Shannon L. Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598, Berlin, Germany. Association for Computational Linguistics.

Honor Hsin, John Torous, and Laura Roberts. 2016. An Adjuvant Role for Mobile Health in Psychiatry. *JAMA Psychiatry*, 73(2):103–104.

Alejandro Interian, Megan Chesin, Anna Kline, Rachael Miller, Lauren St. Hill, Miriam Latorre, Anton Shcherbakov, Arlene King, and Barbara Stanley. 2018. Use of the columbia-suicide severity rating scale (c-ssrs) to classify suicidal behaviors. *Archives of suicide research*, 22(2):278–294.

Shaoxiong Ji, Xue Li, Zi Huang, and Erik Cambria. 2021a. Suicidal ideation and mental disorder detection with attentive relation networks. *Neural Computing and Applications*.

Shaoxiong Ji, Shirui Pan, Xue Li, Erik Cambria, Guodong Long, and Zi Huang. 2021b. Suicidal ideation detection: A review of machine learning methods and applications. *IEEE Transactions on Computational Social Systems*, 8(1):214–226.

John G Keilp, Michael F Grunebaum, Marianne Gorlyn, Simone LeBlanc, Ainsley K Burke, Hanga Galfalvy, Maria A Oquendo, and J John Mann. 2012. Suicidal ideation and the subjective aspects of depression. *Journal of affective disorders*, 140(1):75–81.

Mrinal Kumar, Mark Dredze, Glen Coppersmith, and Munmun De Choudhury. 2015. Detecting changes in suicide content manifested in social media following celebrity suicides. In *Proceedings of the 26th ACM Conference on Hypertext & Social Media*, HT '15, page 85–94, New York, NY, USA. Association for Computing Machinery.

Kathryn P Linthicum, Katherine Musacchio Schafer, and Jessica D Ribeiro. 2019. Machine learning in suicide science: Applications and ethics. *Behavioral sciences & the law*, 37(3):214–222.

Ziyin Liu, Zhikang Wang, Paul Pu Liang, Ruslan Salakhutdinov, Louis-Philippe Morency, and Masahito Ueda. 2019. Deep gamblers: Learning to abstain with portfolio theory. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 10622–10632.

Matthew Matero, Akash Idnani, Youngseo Son, Salvatore Giorgi, Huy Vu, Mohammad Zamani, Parth Limbachiya, Sharath Chandra Guntuku, and H. Andrew Schwartz. 2019. Suicide risk assessment with multi-level dual-context language and BERT. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 39–44, Minneapolis, Minnesota. Association for Computational Linguistics.

William V McCall, Ben Porter, Ashley R Pate, Courtney J Bolstad, Christopher W Drapeau, Andrew D Krystal, Ruth M Benca, Meredith E Rumble, and

Michael R Nadorff. 2021. Examining suicide assessment measures for research use: Using item response theory to optimize psychometric assessment for research on suicidal ideation in major depressive disorder. *Suicide and Life-Threatening Behavior*, 51(6):1086–1094.

Catherine M McHugh, Amy Corderoy, Christopher James Ryan, Ian B Hickie, and Matthew Michael Large. 2019. Association between suicidal ideation and suicide: meta-analyses of odds ratios, sensitivity, specificity and positive predictive value. *BJPsych open*, 5(2).

Rohan Mishra, Pradyumn Prakhar Sinha, Ramit Sawhney, Debanjan Mahata, Puneet Mathur, and Rajiv Ratn Shah. 2019. SNAP-BATNET: Cascading author profiling and social network graphs for suicide ideation detection on social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 147–156, Minneapolis, Minnesota. Association for Computational Linguistics.

John L Oliffe, John S Ogrodniczuk, Joan L Bottorff, Joy L Johnson, and Kristy Hoyak. 2012. "you feel like you can't live anymore": Suicide from the perspectives of canadian men who experience depression. *Social science & medicine*, 74(4):506–514.

Kelly Posner, Gregory K Brown, Barbara Stanley, David A Brent, Kseniya V Yershova, Maria A Oquendo, Glenn W Currier, Glenn A Melvin, Laurence Greenhill, Sa Shen, et al. 2011. The columbia–suicide severity rating scale: initial validity and internal consistency findings from three multisite studies with adolescents and adults. *American journal of psychiatry*, 168(12):1266–1277.

Cornelius Puschman. 2017. Bad judgment, bad ethics? *Internet Research Ethics for the Social Age*, page 95.

The Register. 2020. Researchers made an openai gpt-3 medical chatbot as an experiment. it told a mock patient to kill themselves.

Ramit Sawhney, Harshit Joshi, Saumya Gandhi, and Rajiv Ratn Shah. 2021a. Towards ordinal suicide ideation detection on social media. WSDM '21, page 22–30, New York, NY, USA. Association for Computing Machinery.

Ramit Sawhney, Harshit Joshi, Rajiv Ratn Shah, and Lucie Flek. 2021b. Suicide ideation detection via social and temporal user representations using hyperbolic learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2176–2190, Online. Association for Computational Linguistics.

Han-Chin Shing, Philip Resnik, and Douglas Oard. 2020. A prioritization model for suicidality risk assessment. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8124–8137, Online. Association for Computational Linguistics.

Pradyumna Prakhar Sinha, Rohan Mishra, Ramit Sawhney, Debanjan Mahata, Rajiv Ratn Shah, and Huan Liu. 2019. #suicidal - A multipronged approach to identify and explore suicidal ideation in twitter. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 941–950. ACM.

Dean F Sittig and Hardeep Singh. 2015. A new socio-technical model for studying health information technology in complex adaptive healthcare systems. In *Cognitive informatics for biomedicine*, pages 59–80. Springer.

Takanao Tanaka and Shohei Okamoto. 2021. Increase in suicide following an initial decline during the covid-19 pandemic in japan. *Nature human behaviour*, 5(2):229–238.

WHO. 2019. Suicide data.

Ayah Zirikly, Philip Resnik, Özlem Uzuner, and Kristy Hollingshead. 2019. CLPsych 2019 shared task: Predicting the degree of suicide risk in Reddit posts. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 24–33, Minneapolis, Minnesota. Association for Computational Linguistics.

Liu Ziyin, Blair Chen, Ru Wang, Paul Pu Liang, Ruslan Salakhutdinov, Louis-Philippe Morency, and Masahito Ueda. 2020. Learning not to learn in the presence of noisy labels. *ArXiv*, abs/2002.06541.

When classifying grammatical role, BERT doesn't care about word order…except when it matters

Isabel Papadimitriou
Stanford University
isabelvp@stanford.edu

Richard Futrell
University of California, Irvine
rfutrell@uci.edu

Kyle Mahowald
The University of Texas at Austin
mahowald@utexas.edu

Abstract

Because meaning can often be inferred from lexical semantics alone, word order is often a redundant cue in natural language. For example, the words *chopped*, *chef*, and *onion* are more likely used to convey "The chef chopped the onion," not "The onion chopped the chef." Recent work has shown large language models to be surprisingly word order invariant, but crucially has largely considered natural *prototypical* inputs, where compositional meaning mostly matches lexical expectations. To overcome this confound, we probe grammatical role representation in English BERT and GPT-2, on instances where lexical expectations are not sufficient, and word order knowledge is necessary for correct classification. Such *non-prototypical* instances are naturally occurring English sentences with inanimate subjects or animate objects, or sentences where we systematically swap the arguments to make sentences like "The onion chopped the chef". We find that, while early layer embeddings are largely lexical, word order is in fact crucial in defining the later-layer representations of words in semantically non-prototypical positions. Our experiments isolate the effect of word order on the contextualization process, and highlight how models use context in the uncommon, but critical, instances where it matters.

1 Introduction and Prior Work

Large language models create contextual embeddings of the words in their input, starting with a static embedding of each token and progressively adding more contextual information in each layer (Devlin et al., 2019; Brown et al., 2020; Manning et al., 2020). While these contextual embedding models are often praised for capturing rich grammatical structure, a spate of recent work has shown that they are surprisingly invariant to scrambling word order (Sinha et al., 2021; Hessel and Schofield, 2021; Pham et al., 2021; Gupta et al., 2021; O'Connor and Andreas, 2021) and that grammatical knowledge like part of speech, often attributed to contextual embeddings, is actually also captured by fixed embeddings (Pimentel et al., 2020). These results point to a puzzle: how can syntactic contextual information be important for language understanding when the words themselves, not their order, are what matter?

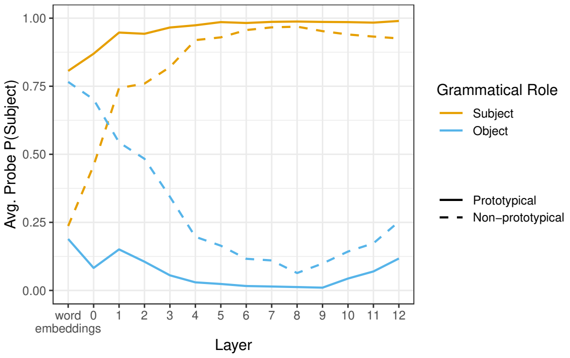


Figure 1: Probabilities of probes trained to differentiate subjects from objects in BERT embeddings. We separate our evaluation examples by prototypicality: whether the ground truth grammatical role is what we would expect given the word out of context. The majority of natural examples are prototypical (solid lines), and so if we average all cases we cannot see that grammatical information is gradually acquired in the first half of the network for cases where lexical information is non-prototypical. The equivalent figures for GPT-2 are in Appendix A.

We argue that this apparent paradox arises because of the redundant structure of language itself. Lexical distributional information alone inherently captures a great deal of meaning (Erk, 2012; Mitchell and Lapata, 2010; Tal and Arnon, 2022), and typically both humans and machines can reconstruct meanings of sentences under local scrambling of words (Mollica et al., 2020; Clouatre et al., 2021). In this paper, we study model behaviour in cases where word order is informative and *is not* redundant with lexical information.

We focus on the feature of **grammatical role**

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 636 - 643
May 22-27, 2022 ©2022 Association for Computational Linguistics

(whether a noun is the subject or the object of a clause). Most natural clauses are **prototypical**: in a sentence like “the chef chopped the onion”, the grammatical roles of *chef* and *onion* are clear to humans from the words alone, without word order or context (see Mahowald et al., 2022, for experiments in English and Russian in which human participants successfully guessed which of two nouns was the subject and which was the object of a simple transitive clause, in the absence of word order and contextual information). This means syntactic word order is often redundant with lexical semantics. Whether hand-constructed or corpus-based, most studies probing contextual representations have used prototypical sentences as input, where syntactic word order may not have much information to contribute to core meaning beyond the words themselves.

Yet human language can use syntax to deviate from the expectations generated by lexical items: we can also understand the absurd meaning of a rare **non-prototypical** sentence like “The onion chopped the chef” (Garrett, 1976; Gibson et al., 2013). Is this use of syntactic word order available to pretrained models? In this paper, we train grammatical role probes on the embedding spaces of BERT and GPT-2[1], and evaluate them on these rare non-prototypical examples, where the meaning of words in context is different from what we would expect from looking at the words alone. We focus on English because grammatical role is directly dependent on word order in English, and because we had access to sufficiently large English parsed corpora such that we could generate non-prototypical sentences, easily check them, and filter to grammatical ones.

We probe for grammatical role because it is key to the basic compositional semantic structure of a sentence (Dixon, 1979; Comrie, 1989; Croft, 2001). While fixed lexical semantics contains information about grammatical role (animate nouns are likely to be subjects, etc), the grammatical role of a word in English is ultimately determined by syntactic word order. Probing grammatical role lets us examine the interplay between syntactic word order and lexical semantics in forming compositional meaning through model layers.

For all of our experiments, we train grammatical role probes with standard data and test them on either prototypical cases or non-prototypical cases (where word order matters), to understand if grammatical embedding under normal circumstances is sensitive to word order. Our experiments reveal three key findings:

1. Lexical semantics plays a key role in organizing embedding space in early layer representations, and non-lexical compositional features are expressed gradually in later layers, as shown by probe performance on non-prototypical sentences (Experiment 1, Figure 1).

2. Embeddings represent meaning that is imparted *only* by syntactic word order, overriding lexical and distributional cues. When we control for distributional co-occurrence factors by evaluating our probes on **argument swapped sentences** (like “The onion chopped the chef”, real sample in Appendix B), probes can differentiate the same word in different roles (Experiment 2, Figure 2).

3. Syntactic word order is significant beyond just local coherence: the compositional information of syntactic word order is lost when we test our probes on locally-shuffled sentences, that keep local lexical coherence but break acute syntactic relations (Figure 3).

More generally, we highlight the importance of examining models using non-prototypical examples, both for understanding the strength of lexical influence in contextual embeddings, but also for accurately isolating syntactic processing where it is taking place.[2]

2 Why non-prototypical probing?

As opposed to more general syntactic probing tasks (e.g., dependency parsing), grammatical role is a linguistically significant yet specific task that is both syntactic *and* semantic. As such, we can choose these linguistically-informed sets of non-prototypical examples where the lexical semantics does not match the compositional meaning implied by the syntax.

Non-prototypical examples give us a unique perspective on how syntactic machinery like word order influences compositional meaning representation *independently* from lexical semantics. Stud-

[1] Results are similar for the two models, so we visualize BERT results here, and include GPT-2 figures in Appendix A.

[2] The code to run our experiments is at `https://github.com/toizzy/except-when-it-matters`

ies in probing have controlled for lexical semantics by substituting content words for nonce words ("jabberwocky" sentences, as in Hall Maudslay and Cotterell, 2021; Goodwin et al., 2020) or random real words ("colorless green idea" sentences, as in Gulordava et al., 2018). A tradeoff is that these methods lead to out-of-distribution sentences whose words are unlikely to ever co-occur naturally. Rather than bleaching the effect of lexical semantics, our setup lets us examine the interplay between lexical semantics and syntactic representation in a controlled environment, isolating the effects of syntactic word order while using in-distribution examples.

Recent work on representation probing has focused on improving probing methodologies to make sure that extracted information is not spurious or not simply lexical (Hewitt and Liang, 2019; Belinkov, 2022; Voita and Titov, 2020; Hewitt et al., 2021; Pimentel et al., 2020). Our experiments are a complementary approach, where we use standard probing methods, but use linguistically-informed *data selection* to address the ambiguity of what classifiers are extracting.

3 Experiment 1: Grammatical Subjecthood Probes

In Experiment 1, we evaluate grammatical role probes on prototypical instances, where grammatical role lines up with lexical expectations, and non-prototypical instances, where it does not.

3.1 Methods

We train a 2-level perceptron classifier probe with 64 hidden units to distinguish the layer embeddings of nouns that are *transitive subjects* from nouns that are *transitive objects*, as in Papadimitriou et al. (2021). We train a separate classifier for each model layer, as well as training a classifier on the static word embedding space of the models without the position embeddings added (before layer 0). The probe classifiers are binary, taking the layer embedding of a noun and predicting whether it is a transitive subject or a transitive object. Probe training data comes from Universal Dependencies treebanks: we pass single sentences from the treebanks through the models, and use dependency annotations to label each layer embedding for whether it represents a transitive subject, a transitive object, or neither (not included in training). The training set is balanced, and consists of 864 embeddings of subject nouns, and 864 embeddings of object nouns. We train all probes for 20 epochs, for consistency. The embedding models that we use are `bert-base-uncased` and `gpt2`. For our analysis, we call a noun a prototypical subject if the probe probability for its word embedding (pre-layer 0) is greater than 0.5, and a prototypical object if it is less.

3.2 Results

Prototypical and non-prototypical arguments differ in probing behavior across layers, as demonstrated in Figure 1. For prototypical instances (solid lines), syntactic information is conflated with type-level information and so probe accuracy is high starting from layer 0 (word embeddings + position embeddings), and stays consistent throughout the network. However, when we look at non-prototypical instances (dashed lines), we see that the embeddings from layer to layer have very different grammatical encodings, with type-level semantics dominating in the early layers and more general syntactic knowledge only becoming extractable by our probes in later layers.

Crucially, since prototypical examples dominate in frequency in any corpus, the average probe accuracy across all examples is high for all layers, and the grammatical encoding of subjecthood, which is accurate only after the middle layers of the model, would be hidden. Separating out non-prototypical examples illustrates how the syntax of a phrase can arise independently from type-level information through transformer layers, while also showcasing the importance of lexical semantics in forming embedding space geometry in the first half of the network.

4 Experiment 2: Controlling for Distributional Information by Swapping Subjects and Objects

In Experiment 1 we show that the contextualization process consists of gradual grammatical information gain for non-prototypical examples, even though this is largely obscured in the majority prototypical examples where the lexical semantics also contains accurate syntactic information. In this experiment, we ask: does this contextualized information about grammatical role stem from word order and syntax, or from distributional (bag-of-words) effects when seeing all words in the sentence? We answer this question by creating example pairs

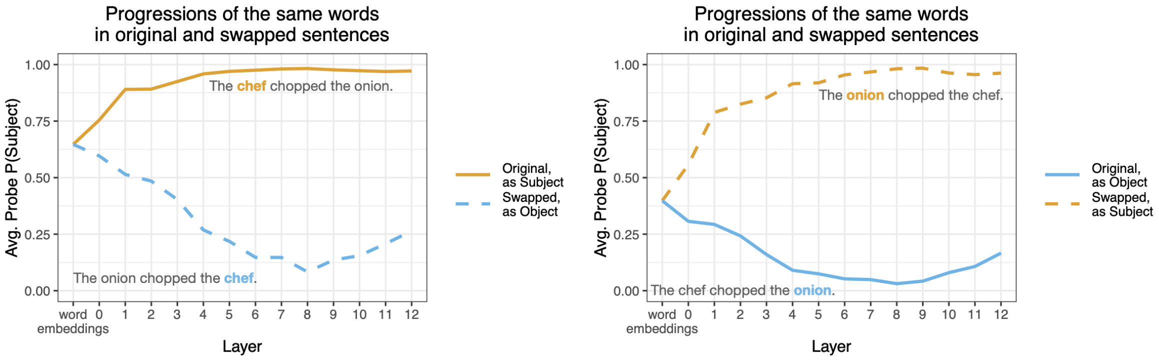


Figure 2: Average probe probabilities for our argument-swapped test set. We visualize the probabilities for the same words in the original treebank sentence (eg. “The chef chopped the onion”, solid lines) and after manual swapping (eg. “The onion chopped the chef”, dashed lines). When probing the geometry of grammatical role, *the same words in the same distributional contexts* are clearly differentiated throughout contextualization in BERT layers, due to the impact of syntactic word order. The figures show the average probe predictions over our whole swapped test set.

where we control for distributional information by keeping all the words the same, but swapping the positions of the subject and the object. Such pairs of the type “The chef chopped the onion” → “The onion chopped the chef” (real sample in Appendix B) have identical distributional information. To accurately classify grammatical role in both sentences, the model we’re probing would have to be attuned to the ways in which small changes in word order globally affect meaning.

4.1 Methods

We use the same probing classifiers from Experiment 1, and evaluate on a special test set of pairs of sentences that have the subject and direct object of one clause swapped. To create the swapped sentences, we search the UD treebank for verbs that have lexical, non-pronoun direct subjects and direct objects, check that the subject and object have the same number (singular or plural), and also check that neither of them are part of a compound word or a flat dependency word that would be separated (like a full name). If a sentence contains a verb where its arguments fulfill all of these requirements, we swap the position of the subject and the object to create a second, swapped sentence, and add the sentence pair (original and swapped) to our evaluation set[3]. A random sample of our swapped sentences is in Appendix B.

4.2 Results

When testing our probes on pairs of normal and swapped sentences, we find that our probes from Experiment 1 correctly classify both the normal and the swapped sentences with high accuracy in higher layers. Since we test our probes on controlled pairs that have the same distributional information, we can isolate effect of syntactic word order in influencing meaning representation. This is demonstrated in Figure 2, where probe predictions for the same set of words in the same distributional context diverges significantly depending on whether the word is in subject or object position. Our results indicate that, separate from distributional effects, models have learned to represent the ways in which syntactic word order can *independently* affect meaning.

4.3 Are these results just due to general position information?

Our results in Experiment 2 indicate that syntactic word order information can affect model representations of word meaning, even when we keep lexical and distributional information constant. A question still remains: does the divergence demonstrated in Figure 2 stem from the fine-grained ways in which word order influences syntax in English, or from heuristics based on primacy (whether a word is earlier or later in a sentence)? To further investigate this, we train and test probes on sentences where word order is locally scrambled so that no word moves more than 2 slots, and so general primacy and local coherence is preserved. As shown in Figure 3, probes trained on these locally shuffled sentences do not fare better than chance

[3] We do not filter for prototypical subjects and objects in this process, since we are assessing the effect of all distributional information: a sentence like “The onion made the chef cry” has nouns in non-prototypical roles, but is still much more felicitous than its swapped version

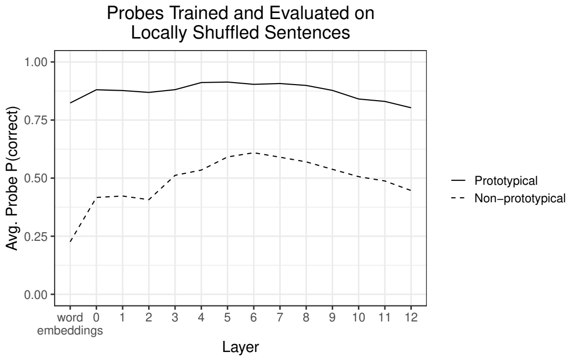


Figure 3: Probe accuracies for sentences where the words have been locally scrambled such that no word moves more than 2 slots. Probe performance for non-prototypical sentences is close to chance, indicating that general positional information (still available after local scrambling) is not enough to recover grammatical role. However, the lexical semantics is preserved through layers in these scrambled instances as evidenced by the steady probe performance on prototypical sentences.

on non-prototypical examples. While prototypical lexical information can aid classification (solid line), general primacy information is not sufficient to overcome lexical cues and cause the word-order-dependent representation we demonstrate in Figure 2.

5 Discussion

While recent work has shown that large language models come to rely largely on distributional semantic information, we consider the model's ability to *overcome* these distributional cues. Research showing that models rely on lexical and distributional information is not at odds with our findings that this can be overridden. In fact, even though humans can accurately understand non-prototypical sentences, human syntactic processing is often influenced by the lexical semantics of words, as evidenced by studies on human subjects (Frazier and Rayner, 1982; Rayner et al., 1983; Ferreira and Henderson, 1990) as well as by lexically-influenced syntactic processes in human languages, like differential object marking (Aissen, 2003)—a phenomenon whereby non-prototypical grammatical objects are marked.

More generally, while we have shown that it is tempting for a straightforward probing approach to conclude that grammatical role information is available to the lowest layers of BERT, separately analyzing prototypical and non-prototypical arguments makes it clear that the picture is more complicated. At lower layers, BERT representations can *typically* classify subjects and objects, but when a non-prototypical meaning is expressed, accurate classification is not available until the higher layers.

We argue that considering probing performance on these non-prototypical instances is crucial. A key design feature of human language is the ability to talk about things that aren't there or don't exist (Hockett, 1960), and it has been argued that the combinatoric power of syntax exists to allow humans to say things that are subtle, surprising, or impossible (Garrett, 1976; Chomsky, 1957). Thus, considering probing accuracy on the *average* task may be misleading. Insofar as being able to understand non-prototypical meanings is a hallmark of human language and insofar as these meanings may differ in systematic ways from prototypical meanings, considering such cases is crucial for understanding how language models represent language.

6 Acknowledgments

This work was supported by National Science Foundation Grants No. 2104995 to KM, No. 1947307 to RF, and a Graduate Research Fellowship to IP. We thank Dan Jurafsky and Adina Williams for helpful discussions, and Kaitlyn Zhou, Dallas Card, and J. Adolfo Hermosillo for comments on drafts.

References

Judith Aissen. 2003. Differential object marking: Iconicity vs. economy. *Natural Language & Linguistic Theory*, 21(3):435–483.

Yonatan Belinkov. 2022. Probing Classifiers: Promises, Shortcomings, and Advances. *Computational Linguistics*, 48(1):1–13.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Noam Chomsky. 1957. *Syntactic structures*. Walter de Gruyter.

Louis Clouatre, Prasanna Parthasarathi, Amal Zouaq, and Sarath Chandar. 2021. Demystifying neural language models' insensitivity to word-order. *arXiv preprint arXiv:2107.13955*.

Bernard Comrie. 1989. *Language Universals and Linguistic Typology*, 2nd edition. University of Chicago Press, Chicago.

William A. Croft. 2001. Functional approaches to grammar. In Neil J. Smelser and Paul B. Baltes, editors, *International Encyclopedia of the Social and Behavioral Sciences*, pages 6323–6330. Elsevier Sciences, Oxford.

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

Robert M. W. Dixon. 1979. Ergativity. *Language*, 55:59–138.

Katrin Erk. 2012. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653.

Fernanda Ferreira and John M Henderson. 1990. Use of verb information in syntactic parsing: Evidence from eye movements and word-by-word self-paced reading. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 16(4):555.

Lyn Frazier and Keith Rayner. 1982. Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences. *Cognitive Psychology*, 14(2):178–210.

Merrill F. Garrett. 1976. Syntactic processes in sentence production. In *New Approaches to Language Mechanisms*, pages 231–255. North-Holland, Amsterdam.

Edward Gibson, Steven T. Piantadosi, Kimberly Brink, Leon Bergen, Eunice Lim, and Rebecca Saxe. 2013. A noisy-channel account of crosslinguistic word-order variation. *Psychological Science*, 24(7):1079–1088.

Emily Goodwin, Koustuv Sinha, and Timothy J. O'Donnell. 2020. Probing linguistic systematicity. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1958–1969, Online. Association for Computational Linguistics.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. 2021. BERT & family eat word salad: Experiments with text understanding. *arXiv preprint arXiv:2101.03453*.

Rowan Hall Maudslay and Ryan Cotterell. 2021. Do syntactic probes probe syntax? Experiments with Jabberwocky probing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 124–131, Online. Association for Computational Linguistics.

Jack Hessel and Alexandra Schofield. 2021. How effective is BERT without word ordering? Implications for language understanding and data privacy. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 204–211.

John Hewitt, Kawin Ethayarajh, Percy Liang, and Christopher Manning. 2021. Conditional probing: measuring usable information beyond a baseline. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1626–1639, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Charles F. Hockett. 1960. The origin of language. *Scientific American*, 203(3):88–96.

Kyle Mahowald, Evgeniia Diachek, Edward Gibson, Evelina Fedorenko, and Richard Futrell. 2022. Grammatical cues are largely, but not completely, redundant with word meanings in natural language. *arXiv preprint arXiv:2201.12911*.

Christopher D Manning, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences*, 117(48):30046–30054.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.

Francis Mollica, Matthew Siegelman, Evgeniia Diachek, Steven T Piantadosi, Zachary Mineroff, Richard Futrell, Hope Kean, Peng Qian, and Evelina Fedorenko. 2020. Composition is the core driver of the language-selective network. *Neurobiology of Language*, 1(1):104–134.

Joe O'Connor and Jacob Andreas. 2021. What context features can transformer language models use? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 851–864, Online. Association for Computational Linguistics.

Isabel Papadimitriou, Ethan A. Chi, Richard Futrell, and Kyle Mahowald. 2021. Deep subjecthood: Higher-order grammatical features in multilingual BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2522–2532, Online. Association for Computational Linguistics.

Thang Pham, Trung Bui, Long Mai, and Anh Nguyen. 2021. Out of Order: How important is the sequential order of words in a sentence in Natural Language Understanding tasks? In *Findings of the Association for Computational Linguistics: ACL-IJCLNP 2021*, pages 1145–1160, Online. Association for Computational Linguistics.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

Keith Rayner, Marcia Carlson, and Lyn Frazier. 1983. The interaction of syntax and semantics during sentence processing: Eye movements in the analysis of semantically biased sentences. *Journal of Verbal Learning and Verbal Behavior*, 22(3):358–374.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shira Tal and Inbal Arnon. 2022. Redundancy can benefit learning: Evidence from word order and case marking. *Cognition*, 224:105055.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

A Figures for GPT-2 Experiments

We ran our experiments on both BERT and GPT-2 embeddings, and both models had similar behaviors that we discuss in the paper. For clarity, figures in the paper only visualize the BERT results, and we're including the GPT-2 versions of those same figures for comparison. Figure 4 shows the GPT-2 results of Figure 1, Figure 5 shows the GPT-2 results of Figure 2, and Figure 6 shows the GPT-2 result of Figure 3.

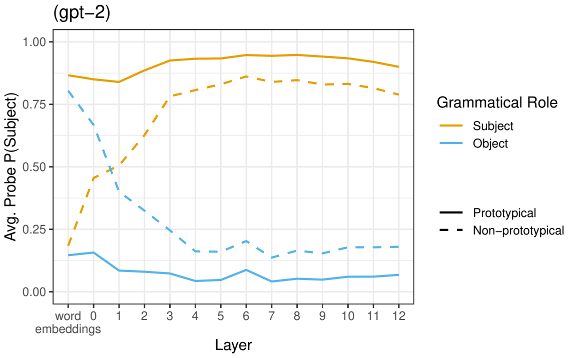


Figure 4: Equivalent to Figure 1 from the main paper, on GPT-2 embeddings

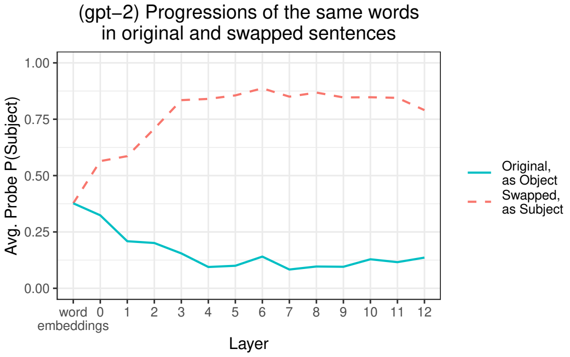


Figure 5: Equivalent to Figure 2 from the main paper, on GPT-2 embeddings. Grammatical representation in GPT-2 embedding also diverges for the same words in the same distributional contexts.

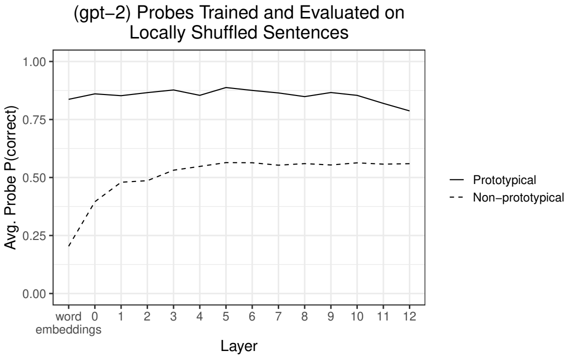


Figure 6: Equivalent to Figure 3 from the main paper, on GPT-2 embeddings. As shown by the dashed line being close to chance, grammatical role information is not extractable from locally shuffled sentences in the non-prototypical cases where lexical semantics do not help

B Sample of argument-swapped sentences

A random sample (not cherry-picked) of our argument-swapped evaluation set, where the subject and the object of clauses are automatically swapped. The original subject is in **bold** and the original object is in ***bold and italics***. The process for creating these sentences is detailed in Section 4.1

On Thursday, with 110 days until the start of the 2014 Winter Paralympics in Sochi, Russia, ***Professor*** interviewed Assistant **Wikinews** in Educational Leadership, Sport Studies and Educational / Counseling Psychology at Washington State University Simon Ličen about attitudes in United States towards the Paralympics.

This ***approach*** shows a more realistic **video** to playing Quidditch.

Second, aggregate ***view*** provides only a high-level **information** of a field, which can make it difficult to investigate causality [23].

A ***hand*** raises her **girl**.

area of the Mississippi River and the destruction of wetlands at its mouth have left the **Alteration** around New Orleans abnormally vulnerable to the forces of nature.

It was known that a moving ***energy*** exchanges its kinetic **body** for potential energy when it gains height.

Thus, when ACPeds issued a statement condemning gender reassignment surgery in 2016 [21], many ***beliefs*** mistook the organization 's political **people** for the consensus view among United States pediatricians — although the peak body for pediatric workers, the American Academy of Pediatrics, has a much more positive view of gender dysphoria [22].

His ***painting*** perfectly combines **art** and Chinese calligraphy.

When the ***inches*** become a few **plants** tall and their leaves mature, it 's time to transplant them to a larger container.

Since the television series' inception, ***reviews*** at The AV Club have written two critical **writers** for each episode:

Triangular Transfer: Freezing the Pivot for Triangular Machine Translation

Meng Zhang, Liangyou Li, Qun Liu
Huawei Noah's Ark Lab
{zhangmeng92, liliangyou, qun.liu}@huawei.com

Abstract

Triangular machine translation is a special case of low-resource machine translation where the language pair of interest has limited parallel data, but both languages have abundant parallel data with a pivot language. Naturally, the key to triangular machine translation is the successful exploitation of such auxiliary data. In this work, we propose a transfer-learning-based approach that utilizes all types of auxiliary data. As we train auxiliary source-pivot and pivot-target translation models, we initialize some parameters of the pivot side with a pre-trained language model and freeze them to encourage both translation models to work in the same pivot language space, so that they can be smoothly transferred to the source-target translation model. Experiments show that our approach can outperform previous ones.

1 Introduction

Machine translation (MT) has achieved promising performance when large-scale parallel data is available. Unfortunately, the abundance of parallel data is largely limited to English, which leads to concerns on the unfair deployment of machine translation service across languages. In turn, researchers are increasingly interested in non-English-centric machine translation approaches (Fan et al., 2021).

Triangular MT (Kim et al., 2019; Ji et al., 2020) has the potential to alleviate some data scarcity conditions when the source and target languages both have a good amount of parallel data with a pivot language (usually English). Kim et al. (2019) have shown that transfer learning is an effective approach to triangular MT, surpassing generic approaches like multilingual MT.

However, previous works have not fully exploited all types of auxiliary data (Table 1). For example, it is reasonable to assume that the source, target, and pivot language all have much monolingual data because of the notable size of parallel data between source-pivot and pivot-target.

approach	X	Y	Z	X-Z	Z-Y	X-Y
no transfer						✓
pivot translation				✓	✓	
step-wise pre-training				✓	✓	✓
shared target transfer	✓		✓		✓	✓
shared source transfer		✓	✓	✓		✓
simple triang. transfer			✓	✓	✓	✓
triangular transfer	✓	✓	✓	✓	✓	✓

Table 1: Data usage of different approaches (Section 3.2). X, Y, and Z represent source, target, and pivot language, respectively. Our triangular transfer uses all types of data.

In this work, we propose a transfer-learning-based approach that exploits all types of auxiliary data. During the training of auxiliary models on auxiliary data, we design parameter freezing mechanisms that encourage the models to compute the representations in the same pivot language space, so that combining parts of auxiliary models gives a reasonable starting point for finetuning on the source-target data. We verify the effectiveness of our approach with a series of experiments.

2 Approach

We first present a preliminary approach that is a simple implementation of our basic idea, for ease of understanding. We then present an enhanced version that achieves better performance. For notation purpose, we use X, Y, and Z to represent source, target, and pivot language, respectively.

2.1 Simple Triangular Transfer

We show the illustration of the preliminary approach in Figure 1, called simple triangular transfer. In Step (1), we prepare a pre-trained language model (PLM) with the pivot language monolingual data. We consider this PLM to define a representation space for the pivot language, and we would like subsequent models to stick to this representation

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 644 - 650
May 22-27, 2022 ©2022 Association for Computational Linguistics

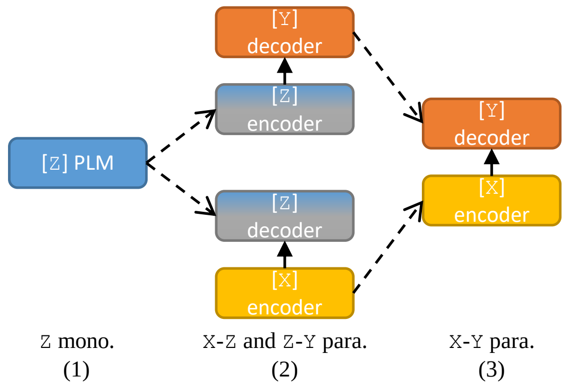


Figure 1: Simple triangular transfer. Dashed lines represent parameter initialization. The gray color within some blocks indicates some parameters are frozen according to the freezing strategy (Section 2.3). Other colors represent trainable parameters in different languages. Below the diagram shows the data used in each step.

space. In order to achieve this, we freeze certain parameters in Step (2) as we train source-pivot and pivot-target translation models, which are partly initialized by the PLM. For example, the pivot-target translation model has the pivot language on the source side, so the encoder is initialized by the PLM, and some (or all) of its parameters are frozen. This ensures that the encoder produces representations in the pivot language space, and the decoder has to perform translation in this space. Likewise, the encoder in the source-pivot translation model needs to learn to produce representations in the same space. Therefore, when the pivot-target decoder combines with the source-pivot encoder in Step (3), they could cooperate more easily in the space defined in Step (1).

We experimented with RoBERTa (Liu et al., 2019) and BART (Lewis et al., 2020) as the PLMs. We found that simple triangular transfer attains about 0.8 higher BLEU by using BART instead of RoBERTa. In contrast, we found that dual transfer (Zhang et al., 2021), one of our baselines, performs similarly with BART and RoBERTa. When used to initialize decoder parameters, RoBERTa has to leave the cross attention parameters randomly initialized, which may explain the superiority of BART for our approach, while dual transfer does not involve initializing decoder parameters. Therefore, we choose BART as our default PLM.

2.2 Triangular Transfer

A limitation of simple triangular transfer is that it does not utilize monolingual data of the source and target languages. A naive way is to prepare source and target PLMs and use them to initialize source-pivot encoder and pivot-target decoder, respectively. However, this leads to marginal improvement for the final source-target translation performance (Section 3.5). This is likely because the source, target, and pivot PLMs are trained independently, so their representation spaces are isolated.

Therefore, we intend to train source and target PLMs in the pivot language space as well. To this end, we design another initialization and freezing step inspired by Zhang et al. (2021), as shown in Figure 2. In this illustration, we use BART as the PLM. Step (2) is the added step of preparing BART models in the source and target languages. As the BART body parameters are inherited from the pivot language BART and frozen, the source and target language BART embeddings are trained to lie in the pivot language space. Then in Step (3), every part of the translation models can be initialized in the pivot language space. Again, we freeze parameters in the pivot language side to ensure the representations do not drift too much.

2.3 Freezing Strategy

There are various choices when we freeze parameters in the pivot language side of the source-pivot and pivot-target translation models. Take the encoder of the pivot-target translation model as the example. In one extreme, we can freeze the embeddings only; this is good for the optimization of pivot-target translation, but may result in a space that is far away from the pivot language space given by the pivot PLM. In the other extreme, we can freeze the entire encoder, which clearly hurts the pivot-target translation performance. This is hence a trade-off. We experiment with multiple freezing strategies between the two extremes, i.e., freezing a given number of layers. We always ensure that the number of frozen layers is the same for the decoder of the source-pivot translation model.

Besides layer-wise freezing, we also try component-wise freezing inspired by Li et al. (2021). In their study, they found that some components like layer normalization and decoder cross attention are necessary to finetune, while others can be frozen. In particular, we experiment with three strategies based on their findings of the most effective ones in their task. These strategies apply to Step (3) of triangular transfer.

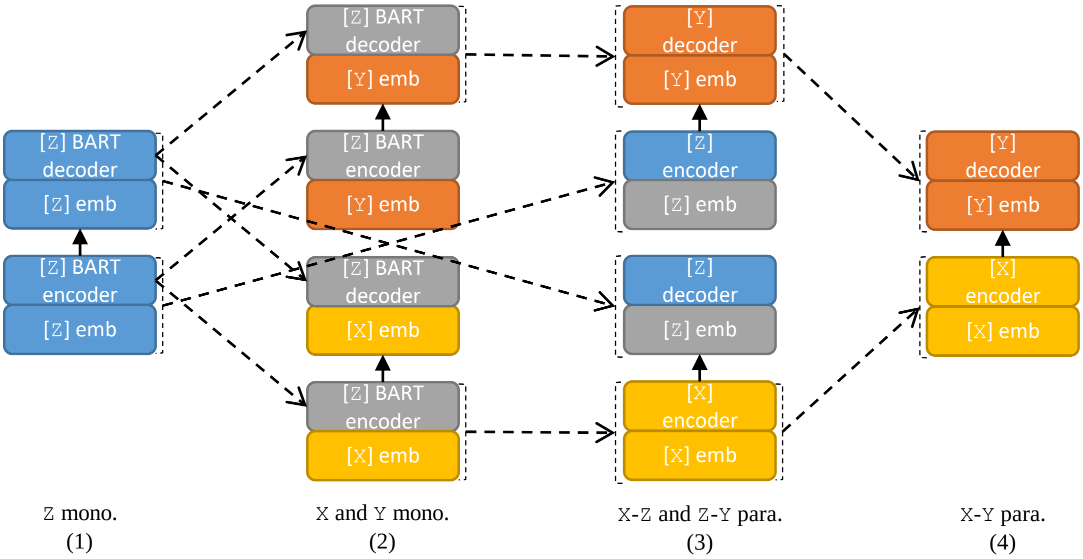


Figure 2: Triangular transfer. Dashed lines represent parameter initialization. The gray color indicates the parameters are frozen. In Step (3) the gray color shows one of the possible freezing strategies (Section 2.3).

language code	# sentence (pair)
En-De	3.1m
Fr-En	29.5m
Fr-De	247k
Zh-En	11.9m
Zh-De	189k
En	93.9m
De	100.0m
Fr	44.6m
Zh	20.0m

Table 2: Training data statistics.

LNA-E,D All layer normalization, encoder self attention, decoder cross attention can be finetuned. Others are frozen.

LNA-D All encoder parameters, decoder layer normalization and cross attention can be finetuned.

LNA-e,D Use LNA-D when training the source-pivot model. When training the pivot-target model, freeze encoder embeddings in addition to LNA-D.

3 Experiments

3.1 Setup

We conduct experiments on French (Fr) → German (De) and Chinese (Zh) → German (De) translation, with English (En) as the pivot language. Training data statistics is shown in Table 2. The evaluation metric is computed by SacreBLEU[1] (Post, 2018). All approaches use Transformer base (Vaswani et al., 2017) as the translation model, but note that pivot translation needs two translation models for decoding, equivalently doubling the number of parameters. Further details can be found in the appendix.

3.2 Baselines

We compare with several baselines as follows.

No transfer This baseline directly trains on the source-target parallel data.

Pivot translation Two-pass decoding by source-pivot and pivot-target translation.

Step-wise pre-training This is one of the approaches in (Kim et al., 2019). It is simple and robust, and has been shown to outperform multilingual MT. It trains a source-pivot translation model and uses the encoder to initialize the encoder of a pivot-target translation model. In order to make this possible, these two encoders need to use a shared source-pivot vocabulary. Then the pivot-target translation model is trained while keeping its encoder frozen. Finally the model is finetuned on source-target parallel data.

Shared target dual transfer Dual transfer (Zhang et al., 2021) is a general transfer learning approach to low-resource machine translation. When

[1] SacreBLEU signature: BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.4.12.

approach	BLEU
no transfer	13.49
pivot translation through no transfer	18.99
step-wise pre-training	18.49
shared target transfer	18.88
shared source transfer	18.89
triangular transfer	19.91

Table 3: Comparison with baselines on Fr→De. Our triangular transfer is significantly better ($p < 0.01$) than baselines by paired bootstrap resampling (Koehn, 2004).

approach	BLEU
no transfer	11.39
pivot translation through no transfer	12.91
triangular transfer	16.03

Table 4: Comparison with no transfer and pivot translation on Zh→De.

applied to triangular MT, it cannot utilize both source-pivot and pivot-target parallel data. Shared target dual transfer uses pivot-target auxiliary translation model and does not exploit source-pivot parallel data.

Shared source dual transfer The shared source version uses source-pivot translation model for transfer and does not exploit pivot-target parallel data.

3.3 Main Results

We present the performance of our approach and the baselines on Fr→De in Table 3. The no transfer baseline performs poorly because it is trained on a small amount of parallel data. The other baselines perform much better. Among them, pivot translation attains the best performance in terms of BLEU, at the cost of doubled latency. Our approach can outperform all the baselines.

Taking pivot translation as the best baseline, we further evaluate our approach on Zh→De. Results in Table 4 show that the performance improvement of our approach is larger for this translation direction.

3.4 The Effect of Freezing Strategies

From Table 5, we can observe the effect of different freezing strategies. For layer-wise freezing, we see a roughly monotonic trend of the Fr-En and En-De performance with respect to the number of frozen layers: The more frozen layers, the lower their BLEU scores. However, the best Fr-De performance is achieved with $L = 3$. This indicates the trade-off between the auxiliary models' performance and the pivot space anchoring. For component-wise freezing, the Fr-En and En-De performance follows a similar trend, but the Fr-De performance that we ultimately care about is not as good.

strategy	Fr-En	En-De	Fr-De
$L = 0$	31.42	20.95	19.62
$L = 1$	31.41	20.98	19.76
$L = 2$	31.55	20.56	19.71
$L = 3$	31.06	20.54	19.91
$L = 4$	30.92	20.22	19.68
$L = 5$	30.39	19.95	19.21
$L = 6$	30.31	19.11	19.02
LNA-E,D	28.72	17.92	17.97
LNA-D	31.08	20.23	18.75
LNA-e,D	31.08	19.97	18.25

Table 5: BLEU scores of different freezing strategies for triangular transfer. For layer-wise freezing, the embeddings and the lowest L layers of the pivot side network are frozen. If $L = 0$, only the embeddings are frozen.

approach	BLEU
pivot translation through no transfer	18.99
pivot translation through BERT2BERT	19.06
shared target transfer	18.88
shared target transfer + naive mono.	18.93
shared source transfer	18.89
shared source transfer + naive mono.	18.97
simple triang. transfer	18.96
simple triang. transfer + naive mono.	19.00
triangular transfer	19.62

Table 6: Naive ways of using auxiliary monolingual data do not bring clear improvement. Our approaches freeze embeddings as the freezing strategy in this table.

3.5 Using Monolingual Data

Table 6 shows the effect of different ways of using monolingual data. The naive way is to prepare PLMs with monolingual data and initialize the encoder or decoder where needed. For pivot translation, this is known as BERT2BERT (Rothe et al., 2020) for the source-pivot and pivot-target translation models. For dual transfer, parts of the auxiliary models can be initialized by PLMs (e.g., for shared target transfer, the pivot-target decoder is initial-

approach	BLEU
no transfer	18.74
shared target transfer	20.53
shared source transfer	20.73
triangular transfer	20.84

Table 7: BLEU scores from training with pivot-based back-translation.

ized). For Step (2) in simple triangular transfer, we can also initialize the pivot-target decoder and source-pivot encoder with PLMs. However, none of the above methods shows clear improvement. This is likely because these methods only help the auxiliary translation models to train, which is not necessary as they can be trained well with abundant parallel data already. In contrast, our design of Step (2) in triangular transfer additionally helps the auxiliary translation models to stay in the pivot language space.

3.6 Pivot-Based Back-Translation

Following Kim et al. (2019), we generate synthetic parallel Fr-De data with pivot-based back-translation (Bertoldi et al., 2008). Specifically, we use a no transfer En→Fr model to translate the English side of En-De data into French, and the authentic Fr-De data are oversampled to make the ratio of authentic and synthetic data to be 1:2. Results in Table 7 show that triangular transfer and dual transfer clearly outperform the no transfer baseline.

4 Conclusion

In this work, we propose a transfer-learning-based approach that utilizes all types of auxiliary data, including both source-pivot and pivot-target parallel data, as well as involved monolingual data. We investigate different freezing strategies for training the auxiliary models to improve source-target translation, and achieve better performance than previous approaches.

References

Nicola Bertoldi, Madalina Barbaiani, Marcello Federico, and Roldano Cattoni. 2008. Phrase-based statistical machine translation with pivot languages. In *Proceedings of the 5th International Workshop on Spoken Language Translation: Papers*, pages 143–149, Waikiki, Hawaii.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Michael Auli, and Armand Joulin. 2021. Beyond English-Centric Multilingual Machine Translation. *Journal of Machine Learning Research*, 22(107):1–48.

Baijun Ji, Zhirui Zhang, Xiangyu Duan, Min Zhang, Boxing Chen, and Weihua Luo. 2020. Cross-lingual Pre-training Based Transfer for Zero-shot Neural Machine Translation. In *AAAI*.

Yunsu Kim, Petre Petrov, Pavel Petrushkov, Shahram Khadivi, and Hermann Ney. 2019. Pivot-based Transfer Learning for Neural Machine Translation between Non-English Languages. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 866–876, Hong Kong, China. Association for Computational Linguistics.

Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xian Li, Changhan Wang, Yun Tang, Chau Tran, Yuqing Tang, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. 2021. Multilingual Speech Translation from Efficient Finetuning of Pretrained Models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 827–838, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on*

Machine Translation: Research Papers, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the Output Embedding to Improve Language Models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

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

Meng Zhang, Liangyou Li, and Qun Liu. 2021. Two Parents, One Child: Dual Transfer for Low-Resource Neural Machine Translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2726–2738, Online. Association for Computational Linguistics.

A Data and Preprocessing

We gather data from WMT and ParaCrawl, shown in Tables 8 and 9.

We use `jieba`[2] for Chinese word segmentation, and Moses[3] scripts for punctuation normalization and tokenization of other languages. The corpora are deduplicated. Each language is encoded with byte pair encoding (BPE) (Sennrich et al., 2016) with 32k merge operations. The BPE codes and vocabularies are learned on each language's monolingual data, and then used to segment parallel data. Sentences with more than 128 subwords are removed. Parallel sentences are cleaned with length ratio 1.5 (length counted by subwords).

B Hyperparameters

Our implementation is based on `fairseq` (Ott et al., 2019). We share decoder input and output embeddings (Press and Wolf, 2017). The optimizer is Adam. Dropout and label smoothing are both set to 0.1. The batch size is 6,144 per GPU and we train on 8 GPUs. The peak learning rate is 5×10^{-4} for the no transfer baseline and auxiliary models, 1×10^{-4} for the Fr→De model of step-wise pre-training and dual transfer, and 7×10^{-5} for the last step of triangular transfer. The learning rate warms up for 4,000 steps, and then follows inverse square root decay. Early stopping happens when the development BLEU does not improve for 10 epochs.

RoBERTa and BART models use exactly the same architecture as Transformer base. The mask ratio is 15%. The batch size is 256 sentences per GPU, and each sentence contains up to 128 tokens. The learning rate warms up for 10,000 steps to the peak 5×10^{-4}, and then follows polynomial decay. They are trained for 125k steps.

We use beam size of 5 for decoding, including for pivot translation and pivot-based back-translation.

[2] `https://github.com/fxsjy/jieba`

[3] `https://github.com/moses-smt/mosesdecoder`

lang.	source	train	dev	test
En-De	WMT 2019	Europarl v9, News Commentary v14, Document-split Rapid corpus	newstest2011	newstest2012
Fr-En	WMT 2015	Europarl v7, News Commentary v10, UN corpus, 10^9 French-English corpus	newstest2011	newstest2012
Fr-De	WMT 2019	News Commentary v14, newstest2008-2010	newstest2011	newstest2012
Zh-En	ParaCrawl	ParaCrawl v9	newsdev2017	newstest2017
Zh-De	WMT 2021	News Commentary v16 - dev - test	3k split	3k split

Table 8: Parallel data source.

lang.	source	name
En	WMT 2018	News Crawl 2014-2017
De	WMT 2021	100m subset from WMT 2021
Fr	WMT 2015	Europarl v7, News Commentary v10, News Crawl 2007-2014, News Discussions
Zh	WMT 2021	News Crawl, Zh side of parallel data

Table 9: Monolingual data source.

Can Visual Dialogue Models Do Scorekeeping? Exploring How Dialogue Representations Incrementally Encode Shared Knowledge

Brielen Madureira **David Schlangen**
Computational Linguistics
Department of Linguistics
University of Potsdam, Germany
{madureiralasota, david.schlangen}@uni-potsdam.de

Abstract

Cognitively plausible visual dialogue models should keep a *mental scoreboard* of shared established facts in the dialogue context. We propose a theory-based evaluation method for investigating to what degree models pretrained on the VisDial dataset incrementally build representations that appropriately do *scorekeeping*. Our conclusion is that the ability to make the distinction between shared and privately known statements along the dialogue is moderately present in the analysed models, but not always incrementally consistent, which may partially be due to the limited need for *grounding interactions* in the original task.

1 Introduction

"There's a cute dog outside!" you say on the phone to your friend. "Sweet. What colour is the dog?", they say. "What dog?" you reply – and your friend is rightfully confused. With your first utterance, you have committed yourself to there being a dog; a commitment you can't just simply ignore later on. Models of dialogue from linguistics and psycholinguistics take this process of *grounding* or *scorekeeping*—making propositions mutual knowledge—to be an elementary fact about dialogue (Lewis, 1979; Clark and Brennan, 1991).

In this short paper, we investigate whether recent NLP models of visual dialogue capture this process. Specifically, we use the VisDial dataset (Das et al., 2017a), which consists of dialogues in English about an image in an asymmetric setting similar to that from the first paragraph, and derive from it diagnostic propositions that should be considered mutual knowledge at a given point in the dialogue, and others whose truth value is only known to one participant at the given time. We then probe dialogue representations built by models pretrained on the VisDial task for whether they correctly track the participants' knowledge and commitments.

2 Related Literature

Representing dialogue context implicitly as the continuous hidden states of neural networks trained in an end-to-end fashion has been a prevailing practice since the works of Vinyals and Le (2015), Sordoni et al. (2015) and Serban et al. (2016). This paradigm also enables multimodal input like images to be easily integrated (Shekhar et al., 2019b). However, there is evidence that the human ability of *collaborative grounding* still lacks in such models, in part due to the limitations of training regimes and datasets (Benotti and Blackburn, 2021).

We witness extensive efforts to look into how these models encode and make use of dialogue history, capture salient information and produce visually grounded representations (Sankar et al., 2019; Agarwal et al., 2020; Greco et al., 2020a,b). The analysis and evaluation of current dialogue models (as Hupkes et al. (2018a), Shekhar et al. (2019a), Parthasarathi et al. (2020), Saleh et al. (2020), Wu and Xiong (2020), *inter alia*) often rely on diagnostic classifiers (Hupkes et al., 2018b) and probing tasks (Belinkov and Glass, 2019), common tools to examine whether representations built by neural networks encode linguistic information.

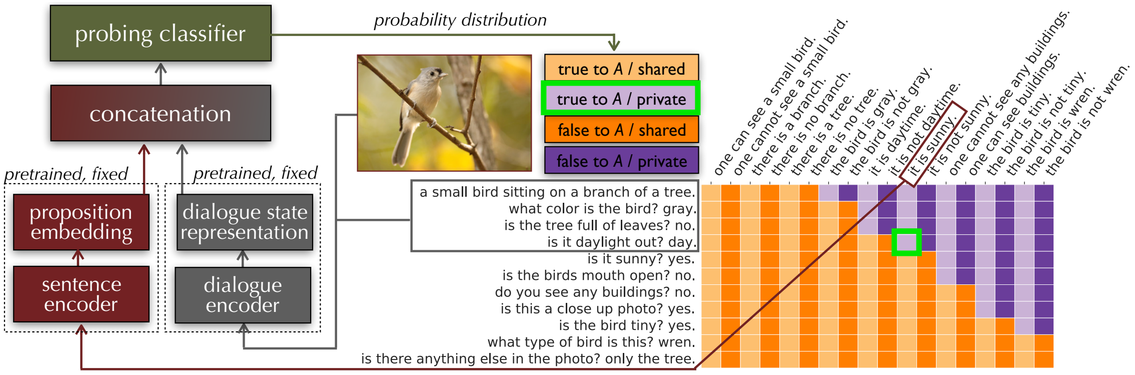
Another purposeful area of research on dialogue revolves around inference. Zhang and Chai (2009, 2010) discuss *conversation entailment*, *i.e.* determining whether a conversation discourse entails a hypothesis. Annotating or generating entailments, contradictions and neutral statements in dialogue datasets is usual in recent works (Welleck et al., 2019; Dziri et al., 2019; Galetzka et al., 2021).

With insights from these three pillars, we propose a probing task for *scorekeeping* (Lewis, 1979) on visual dialogues, formalised in the next section.

3 Problem Statement

Based on the premise that humans keep a *mental scoreboard* of presupposed propositions and per-

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 651 - 664
May 22-27, 2022 ©2022 Association for Computational Linguistics



a small bird sitting on a branch of a tree.
what color is the bird? gray.
is the tree full of leaves? no.
is it daylight out? day.
is it sunny? yes.
is the birds mouth open? no.
do you see any buildings? no.
is this a close up photo? yes.
is the bird tiny? yes.
what type of bird is this? wren.
is there anything else in the photo? only the tree.

one can see a small bird.
one cannot see a small bird.
there is a branch.
there is no branch.
there is a tree.
there is no tree.
the bird is gray.
the bird is not gray.
it is daytime.
it is not daytime.
it is sunny.
it is not sunny.
one cannot see any buildings.
one can see buildings.
the bird is tiny.
the bird is not tiny.
the bird is wren.
the bird is not wren.

Figure 1: A scoreboard representation with generated propositions for a dialogue and architecture of the classifier. It represents the proposition *it is sunny* being correctly classified as (true to A, private) at turn 3. From VisDial training set, ID 8778 (CC-BY 4.0), photo 176904 from MS COCO dataset, ↪Tufted Titmouse by Matt Tillett (CC-BY 2.0).

missible courses of action as a function of what has been stated in a conversation (Lewis, 1979) and on the public/private dichotomy discussed in Ginzburg (2012), we propose a formalisation for the "kinematics of scorekeeping" (Lewis, 1979) on VisDial.

Each dialogue in the VisDial dataset is a tuple $D = (I, Q, A, T, P)$ representing an interaction between a questioner Q and an answerer A. They exchange turns T, which establish propositions P, about a scene depicted in an image I. A sees I, but Q does not. Both are provided with a caption K, which for simplicity we take to be the first turn of A, $t_0 = K$; other turns comprise a question and an answer, $t_i = (q_i, a_i)$, so that $T = (t_i)_{i=0}^{10}$ (as dialogues have 10 turns).[1]

We assume that: i) A does not lie about their interpretation of the image; ii) Q does not ask redundant questions; and iii) a fact disclosed by A immediately becomes a shared commitment, even though in reality this is not always the case (*e.g.* when a misunderstanding happens). Under these assumptions, each t_i discloses a new fact p^i (and its implications) about A's judgement of the image that was unknown to Q until t_{i-1}. P is then defined as a set of N propositions $\{p_1^i, p_2^i, \cdots, p_N^i\}$. Each p_j^i is either the direct entailment of t_i (that is, the expressed proposition), which is established by A to be *true*, or its negation, which is established by A to be *false*. The truth value of p_j^i is known to A throughout the dialogue, but only *privately* so for all $k < i$. It becomes *shared* between A and Q at $k = i$ and remains so until the end of the dialogue.[2]

With this in place, A's scoreboard of a dialogue can be represented by a matrix S_D with dimensions $|T| \times |P|$. Each element $s_{m,n}$ is a tuple $c \in C = \{$(true to A, private), (true to A, shared), (false to A, private), (false to A, shared)$\}$ representing the 'score' of proposition p_n at turn t_m as a class, like the example in Figure 1. Hence, the negation of a fact that A considers true but has not been mentioned yet is labelled as (false to A, private).[3] That way, the scoreboard at a given turn t is given by the t-th row in S and the whole matrix helps visualising how the scoreboard is incrementally updated throughout D.

Probing Task and Model. We design a classification task to examine whether the continuous representations of pretrained visual dialogue models incrementally encode information about the scoreboard represented by S. The probing classifier is a function $f : P_D \times R_{D,t} \to C$, where P_D is the set of propositions in a dialogue D, R is the space of hidden representations of a visual dialogue encoder and C are the scoreboard classes. Based on the probing classifier architecture in Hewitt and Liang (2019), we approximate f as a neural network which maps a dialogue representation r concatenated to a continuous representation z of a proposition to a vector v with a probability distribution over classes, $v = softmax(W_2\sigma(W_1[r; z]))$ (bias term omitted), as illustrated in Figure 1. The class is then predicted with the *argmax* function.

4 Data

Visual Dialogues and Encoders. We use the VisDial dataset v.1.0 (Das et al., 2017a) and the three Q and A encoders (RL_DIV, SL and ICCV_RL)

[1] Except on VisDial test set, where $T < 10$.

[2] Although the set of statements about an image can be infinitely large, we limit P to a finite set here by only considering explicitly disclosed facts (and their negation).

[3] The scoreboard for Q is analogous, except that it cannot differentiate the true/false dimension of private propositions.

from Das et al. (2017b) and Murahari et al. (2019). The first work implemented an end-to-end model to train A and Q using reinforcement learning. The latter is a follow-up study that adds an auxiliary objective function to encourage Q to ask more diverse questions.[4] The VisDial training set contains images from the MS COCO dataset (Lin et al., 2014). Proposition embeddings z are built with Sentence-Transformers (Reimers and Gurevych, 2019).

Generating Probes. The sets P_D are programmatically generated by manipulating QA pairs using rules that identify common lexical and syntactic patterns in VisDial, in a similar fashion as Demszky et al. (2018) and Ribeiro et al. (2019). Whenever the pattern of a QA pair matches a rule, a *direct entailment* and a *direct contradiction* are generated, as those shown in Figure 1.[5]

Dataset Construction. We retrieve the pre-trained dialogue context representations $R_D = \{r_l | 0 \leq l \leq 10\}$, where r_l is the hidden state of the encoder after it processed the dialogue up to turn l in T (and the image and next question for A). We then pair elements in R_D with the embeddings of the generated propositions p^i_j in P_D, forming tuples $\{(r_l, p^i_j) | 0 \leq l \leq 10, 1 \leq j \leq N\}$ which are mapped to the corresponding class $c \in C$. The *true to* A or *false to* A status of a proposition p^i_j remains fixed for all turns in D, since it refers to a fact (according to A's beliefs) about the image, while the *private* status holds for $(r_0, p^i_j), \dots, (r_{i-1}, p^i_j)$ and shifts to *shared* for $(r_i, p^i_j), \dots, (r_{10}, p^i_j)$. The probing dataset is thus composed of datapoints $(r, p, c)_D$ for all D, for all turns' representations $r \in R_D$, for all $p \in P_D$. Propositions generated from captions are downsampled because they outnumber the other turns, resulting in too many propositions that are always shared. In order to avoid bias with respect to the true/false dimension, we sample the training set of propositions enforcing that each type appears as true to A exactly the same number of times as it does as false to A in different dialogues. Table 1 presents a summary (see Appendix for details).

5 Experiments

We train and test the classifier varying three aspects: i) A or Q, ii) main task with all classes in C (TFxPS), plus three variations with reduced dimensions: Only true/false (TF), only private/shared (PS) and merging true/false on the private cases only (PxTSFS) and iii) control tasks (Hewitt and Liang, 2019) (a) replacing r by a random vector (b) replacing r by a null vector, both only on the training set, to quantify how much information can be extracted from propositions alone during training.

	train	valid	test
dialogues	95,369	1,979	6,880
propositions	344,988	23,060	44,954
proposition types	27,011	12,048	19,183
datapoints	3,794,868	253,660	312,102
vocab size	2,709	2,168	2,922
avg. $\|P_D\|$	3.61	11.65	6.53
true to A and private	26.12	22.94	21.42
true to A and shared	23.87	27.05	28.57
false to A and private	26.08	22.94	21.42
false to A and shared	23.91	27.05	28.57

Table 1: Summary of the constructed datasets (after balancing the training set) and proportion of each class.

Evaluation. Results are evaluated with accuracy on class predictions. To avoid any influence that knowing the position in the dialogue could have (early in the dialogue, propositions have a greater chance of being *private*, and vice versa), we evaluate the results at turn 5 (at which there is a more balanced chance of a fact having been mentioned or not). For the error analysis, we reconstruct complete predicted scoreboards and evaluate incremental aspects: In each column, only one shift from private to shared should occur at the right turn (except for caption propositions, which are always shared) and the true/false status should not change.

Implementation. The classifier is implemented with PyTorch (Paszke et al., 2019) and trained with gradient descent using Adam optimizer (Kingma and Ba, 2014) to minimize cross entropy.[6]

6 Results

Table 2 presents the accuracy of all models and tasks at turn 5. The performance on the main task is very similar across encoders, with differences lower than 1.5%. Q outperforms A in all models in the main task. While this is expected, since Q's representations must only keep track of the dialogue whereas A must interpret the image, the difference is only marginal.

[4] Code and model checkpoints available under a BSD license at https://github.com/vmurahari3/visdial-diversity.

[5] The rule-based approach can only generate subsets of the theoretical P_D, but in enough number for the probing task. See Appendix for details and examples.

[6] See Appendix for hyperparameters, model configurations and details on reproducibility. Our code and documentation are available at https://github.com/briemadu/scorekeeping.

task		TFxPS			TF			PS			PxTSFS		
model		(a)	(b)	(c)	(a)	(b)	(c)	(a)	(b)	(c)	(a)	(b)	(c)
A	main	61.80	62.37	61.31	73.05	72.50	72.41	77.29	77.31	77.13	65.57	65.49	65.83
	random r	35.25	37.52	36.60	52.25	52.01	53.17	64.59	68.52	64.07	35.46	39.22	37.48
	null r	37.43	37.19	37.42	50.65	50.65	50.67	62.79	62.85	62.66	37.36	37.51	37.35
Q	main	-	-	-	-	-	-	78.36	79.31	79.21	66.87	65.65	66.38
	random r	-	-	-	-	-	-	60.44	60.53	61.43	35.49	34.58	34.86
	null r	-	-	-	-	-	-	62.42	62.38	62.50	37.28	37.15	37.11

Table 2: Accuracy on test set at turn 5 (32,360 datapoints) for models (a) RL_DIV, (b) SL, (c) ICCV_RL. TFxPS and TF are not applicable to Q because it has no information to distinguish between what A considers true or false on the private dimension. The hypothesis that results of control tasks do not differ from their corresponding main task is rejected for all cases using paired approximate permutation tests with 1,000 shuffles (p-value< 0.01).

For the TF task, the performance on the control tasks is close to random, as expected, but it is higher than random for other tasks. We notice that, while the training dataset is constructed to be balanced in the true/false dimension, information on the private/shared dimension has an inherent bias that is more complex to counterbalance on the training set. Despite the fact that datapoints in the private class do not substantially outnumber the shared class, we observe that each proposition type can have a tendency to occur either early or late in the dialogue (examples in Figure 2), causing them to have an individual skewed distribution towards shared or private at turn 5. This information leak can be used as a shortcut by the classifier.[7] Still, A and Q's representations lead to performances between 8% and 32% higher than the control tasks in all cases.

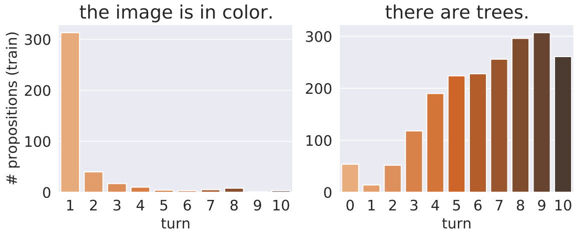


Figure 2: Examples of skewed distributions over dialogue turns which can introduce bias on the private/shared dimension.

Human Performance. Table 3 shows the human performance, estimated as the average accuracy of 3 annotators (0.86 Fleiss' κ on TFxPS) on a sample of 94 datapoints, each from a different dialogue in the test set (not only at turn 5). We observe that humans agree most of the times on their judgements and all models perform well below human level.

task		TFxPS	TF	PS	PxTSFS
human		91.84	94.32	97.51	96.09
A	RL_DIV	52.12	65.95	74.46	65.95
	SL	50.00	72.34	73.40	68.08
	ICCV_RL	52.12	71.27	77.65	67.02
Q	RL_DIV	-	-	75.53	62.76
	SL	-	-	79.78	70.21
	ICCV_RL	-	-	75.53	68.08

Table 3: Accuracy of human judgement compared to the models on a sample (n=94, not only at turn 5).

Error Analysis. We conduct an error analysis on A, main task, TFxPS. The confusion matrix in Figure 3 shows that it is easier to distinguish between true/false to A in the shared dimension, which can be a sign that dialogue information is more salient in the representations than the image.

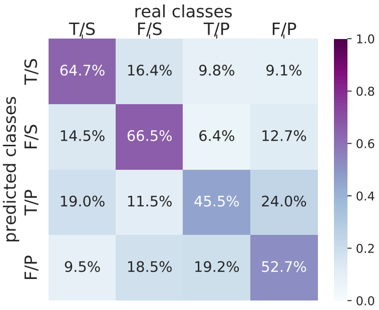


Figure 3: Confusion matrix of predictions at turn 5.

The accuracy on all datapoints with proposition types that occur on the training set is 67.69, higher than for those that do not, which is 53.11.

When we reconstruct full predicted scoreboards, some qualitative shortcomings become evident. A shift from private to shared is predicted at the correct turn for 60.32% of the propositions but only

[7] As pointed by one of the reviewers, this may not be a shortcoming, since it is how dialogue works and humans are probably also exploiting this.

38.24% shifts *only* at the correct turn. Besides, only 44.50% of the propositions have stable predictions regarding the true/false to A dimension.

Figure 4 shows types of errors in the predictions (the Appendix has more examples). We see the same truth value assigned to opposite propositions, the same proposition classified both as true and false at different turns, as well as an occasional oscillation between private/shared throughout the dialogue. These are indications that, although accuracy per label is generally high, the representations do not seem to always allow incrementally stable and consistent predictions throughout the dialogue.

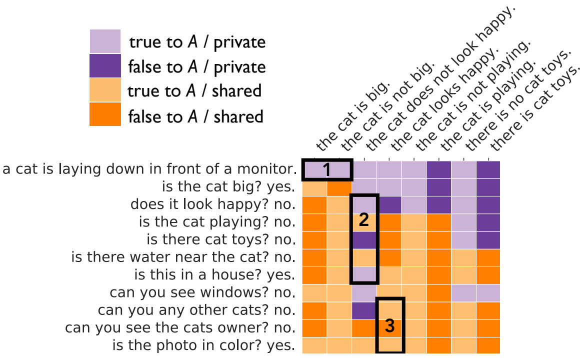


Figure 4: A portion of a predicted scoreboard with some highlighted errors: 1) the same truth value on opposite propositions, 2) oscillation between private and shared, 3) opposite truth values on the same proposition.

7 Scope and Limitations

The results on this paper comprise three visual dialogue models trained using a similar setting on the same dataset. The preprocessing steps used by these models replace some tokens by a UNK token and truncate long captions, which prevents some information to become shared as assumed. Further investigation with other models and data is necessary in future research in order to support more general conclusions. The results also rely on the capabilities of the classifier. Although we performed hyperparameter search, the probing classifier does not completely overfit the full training dataset, thus other architectures and hyperparatemeters can be further investigated.

The rule-based generation of propositions has limitations. It cannot generate propositions for all QA pairs and some rules end up not always yielding grammatically valid sentences, for instance because of countable/uncountable nouns, detection of singular/plural forms and mistakes and typos deriving from the dialogues themselves. Besides, spurious patterns deriving from the implemented rules or other confounds and inherent biases (*e.g.* Figure 2) may exist and be predictive of the classes, which could be captured by the probing classifier and influence (likely overestimating) the results. Enforcing a balance on the training set in terms of true/false to A solves one source of bias but causes its distribution to differ from the validation and test set. The test set also has a different distribution because of its varying number of turns.

Finally, while the assumptions proposed in Section 3 are necessary idealizations for using VisDial for this task, they simplify essential aspects of dialogues, *e.g.* the uncertainty about a fact actually being shared, memory limitations and the many kinds of inference that are used in the accommodation of shared knowledge, such as presuppositions, implicatures, entailments and implicit information. Our method cannot capture background knowledge not explicitly stated in dialogue turns.[8]

8 Conclusion

We have proposed a novel way to do theory-based evaluation of visual dialogue models. Using diagnostic propositions, we investigated to what degree neural network visual dialogue models incrementally build up representations that are appropriate to do *scorekeeping* of shared commitments throughout a dialogue. The evaluated models trained on VisDial capture part of this process, but not always consistently, possibly because this ability is not an elementary component of the training regime. The relatively impoverished nature of the original task in terms of *coordination phenomena* can also limit the capability of models to build good dialogue representations (Schlangen, 2019). Future work should extend the evaluation to other models and reflect on how better and ecologically valid diagnostic datasets for visual dialogues can be constructed.

9 Ethical Considerations

Propositions are direct manipulations of QA pairs and thus reflect the subjective judgments of VisDial crowdworkers. Therefore, they are not *per se* necessarily *true* or *false* with respect to the image, but with respect to A's interpretation expressed as answers. Inappropriate content on images, captions and dialogues can be replicated by the rule-based

[8] We thank the reviewers for pointing out some of the limitations discussed in this section.

proposition generation. To try to remedy this, we filtered out dialogues containing words that could be used for sensitive content. Despite our efforts, we cannot guarantee that we could remove everything, given the size of the dataset and the inherent bias of how humans interpret images. As a result, the only purpose of the propositions is performing the evaluation as proposed here.

Acknowledgements

We are thankful to the anonymous reviewers for their feedback and suggestions, to Wencke Liermann for implementing the interface for the human evaluation and to the student assistants of the Computational Linguistics Lab who contributed on the experiment.

References

Shubham Agarwal, Trung Bui, Joon-Young Lee, Ioannis Konstas, and Verena Rieser. 2020. History for visual dialog: Do we really need it? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8182–8197, Online. Association for Computational Linguistics.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Luciana Benotti and Patrick Blackburn. 2021. Grounding as a collaborative process. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 515–531, Online. Association for Computational Linguistics.

Yonatan Bitton, Gabriel Stanovsky, Roy Schwartz, and Michael Elhadad. 2021. Automatic generation of contrast sets from scene graphs: Probing the compositional consistency of GQA. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 94–105, Online. Association for Computational Linguistics.

Herbert H Clark and Susan E Brennan. 1991. Grounding in communication. In *Perspectives on socially shared cognition.*, pages 127–149. American Psychological Association.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. 2017a. Visual dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 326–335.

Abhishek Das, Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. 2017b. Learning cooperative visual dialog agents with deep reinforcement learning. In *Proceedings of the IEEE international conference on computer vision*, pages 2951–2960.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *arXiv preprint arXiv:1809.02922*.

Nouha Dziri, Ehsan Kamalloo, Kory Mathewson, and Osmar Zaiane. 2019. Evaluating coherence in dialogue systems using entailment. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3806–3812, Minneapolis, Minnesota. Association for Computational Linguistics.

Fabian Galetzka, Jewgeni Rose, David Schlangen, and Jens Lehmann. 2021. Space efficient context encoding for non-task-oriented dialogue generation with graph attention transformer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7028–7041, Online. Association for Computational Linguistics.

Jonathan Ginzburg. 2012. *The Interactive Stance.* Chapter 4: Basic Interaction in Dialogue. Oxford University Press.

Claudio Greco, Alberto Testoni, and Raffaella Bernardi. 2020a. Grounding dialogue history: Strengths and weaknesses of pre-trained transformers. In *International Conference of the Italian Association for Artificial Intelligence*, pages 263–279. Springer.

Claudio Greco, Alberto Testoni, and Raffaella Bernardi. 2020b. Which turn do neural models exploit the most to solve GuessWhat? Diving into the dialogue history encoding in transformers and lstms. In *Proceedings of the 4th Workshop on Natural Language for Artificial Intelligence (NL4AI 2020) co-located with the 19th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2020), Anywhere, November 25th-27th, 2020*, volume 2735 of *CEUR Workshop Proceedings*, pages 29–43. CEUR-WS.org.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Dieuwke Hupkes, Sanne Bouwmeester, and Raquel Fernández. 2018a. Analysing the potential of seq-to-seq models for incremental interpretation in task-oriented dialogue. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 165–174, Brussels, Belgium. Association for Computational Linguistics.

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. 2018b. Visualisation and diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research*, 61:907–926.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

David Lewis. 1979. Scorekeeping in a language game. In *Semantics from different points of view*, pages 172–187. Springer.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Sharid Loáiciga, Simon Dobnik, and David Schlangen. 2021. Reference and coreference in situated dialogue. In *Proceedings of the Second Workshop on Advances in Language and Vision Research*, pages 39–44, Online. Association for Computational Linguistics.

Vishvak Murahari, Prithvijit Chattopadhyay, Dhruv Batra, Devi Parikh, and Abhishek Das. 2019. Improving generative visual dialog by answering diverse questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1449–1454.

Prasanna Parthasarathi, Joelle Pineau, and Sarath Chandar. 2020. How to evaluate your dialogue system: Probe tasks as an alternative for token-level evaluation metrics. *arXiv preprint arXiv:2008.10427*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

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

Marco Tulio Ribeiro, Carlos Guestrin, and Sameer Singh. 2019. Are red roses red? evaluating consistency of question-answering models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6174–6184, Florence, Italy. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.

Abdelrhman Saleh, Tovly Deutsch, Stephen Casper, Yonatan Belinkov, and Stuart Shieber. 2020. Probing neural dialog models for conversational understanding. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 132–143, Online. Association for Computational Linguistics.

Chinnadhurai Sankar, Sandeep Subramanian, Chris Pal, Sarath Chandar, and Yoshua Bengio. 2019. Do neural dialog systems use the conversation history effectively? an empirical study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 32–37, Florence, Italy. Association for Computational Linguistics.

David Schlangen. 2019. Grounded agreement games: Emphasizing conversational grounding in visual dialogue settings. *arXiv cs.CL preprint arXiv:1908.11279*.

Iulian Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Ravi Shekhar, Sandro Pezzelle, Yauhen Klimovich, Aurélie Herbelot, Moin Nabi, Enver Sangineto, and Raffaella Bernardi. 2017. FOIL it! find one mismatch between image and language caption. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 255–265, Vancouver, Canada. Association for Computational Linguistics.

Ravi Shekhar, Ece Takmaz, Raquel Fernández, and Raffaella Bernardi. 2019a. Evaluating the representational hub of language and vision models. In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*, pages 211–222, Gothenburg, Sweden. Association for Computational Linguistics.

Ravi Shekhar, Aashish Venkatesh, Tim Baumgärtner, Elia Bruni, Barbara Plank, Raffaella Bernardi, and Raquel Fernández. 2019b. Beyond task success: A closer look at jointly learning to see, ask, and GuessWhat. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2578–2587, Minneapolis, Minnesota. Association for Computational Linguistics.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 196–205, Denver, Colorado. Association for Computational Linguistics.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. In *ICML Deep Learning Workshop*.

Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2019. Dialogue natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3731–3741, Florence, Italy. Association for Computational Linguistics.

Chien-Sheng Wu and Caiming Xiong. 2020. Probing task-oriented dialogue representation from language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5036–5051, Online. Association for Computational Linguistics.

Chen Zhang and Joyce Chai. 2009. What do we know about conversation participants: Experiments on conversation entailment. In *Proceedings of the SIGDIAL 2009 Conference*, pages 206–215, London, UK. Association for Computational Linguistics.

Chen Zhang and Joyce Chai. 2010. Towards conversation entailment: An empirical investigation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 756–766, Cambridge, MA. Association for Computational Linguistics.

Appendix

A Generating Propositions and Constructing the Datasets

This section presents details about the procedure to turn QA pairs from the VisDial dataset[9] into propositions.

Solving Pronouns. Coreference resolution is specially challenging on visual dialogues, as discussed in Loáiciga et al. (2021). Despite the limitations, we used the model proposed in Lee et al. (2018) to replace pronouns (those that were detected and solved) by their corresponding entity as follows:

1. Merged caption and QA pairs into a single string.
2. Passed string to coreference resolution model to get coreference clusters.[10]
3. Assumed that the first element in the cluster was the entity (its first mention).
4. For each dialogue, checked which questions and answers contained pronouns of interest (he, she, it, they, his, her, its, their, him, them, hers, theirs, this, that, these, those) and replaced them with their corresponding cluster entity, if detected. Assumed the pronoun her was always possessive.
5. If the entity comprised more than N=5 tokens, we did not replace it (because entities spanning over many tokens are very likely to be long portions of the caption that result in wrong propositions).
6. With postprocessing steps, put string back into VisDial format.

On average, 2.24 pronouns were replaced per dialogue on the training set, 2.43 on the validation set and 1.15 on the test set.

Generating Propositions. Automatic generation of diagnostic datasets or adversarial examples via programmatic manipulation rules or templates is a usual step in probing studies, *e.g.* Johnson et al. (2017), Shekhar et al. (2017), Ribeiro et al. (2018) and Bitton et al. (2021). The main steps to turn QA pairs into propositions were to some extent based on Ribeiro et al. (2019) and Demszky et al. (2018). We analysed common patterns of questions

[9] Available at https://visualdialog.org/

[10] Implementation by AllenNLP, version 2.1.0, at https://demo.allennlp.org/coreference-resolution with their pretrained model coref-spanbert-large-2021.03.10.

and answers on VisDial and implemented 34 rules that create entailments and contradictions. Some rules are lexical (*e.g.* questions starting with '*what color is*' and whose answer has a color name) and others depend on POS tag patterns extracted using SpaCy v.3.0.5.[11] Most rules work for polar questions, some work for other types of questions. We noticed that some images and dialogues on VisDial contain inappropriate content. To avoid replicating this on the propositions, we filtered out dialogues that contain words that may be sensitive (see code documentation for details). Propositions were then generated as follows:

1. Parsed the caption to extract nouns and adjectives and generated caption propositions.
2. For each turn, checked whether it matched a manipulation rule.
3. Every rule, when they were applied, generated a direct entailment and a direct contradiction (negation of the entailment).
4. Propositions that contained pronouns (for cases in which coreference resolution did not work), except for *it*, or that were too long (more then 15 tokens) were excluded.

The code documentation has a more detailed description of the rules. The next sections present details of the resulting proposition sets. Note that the number of dialogues in each set is smaller than in the VisDial original splits, because some were filtered out and others had no propositions.

Propositions have four attributes: i) kind of manipulation rule; ii) dialogue and turn from which it derives; iii) a true/false status with respect to what A thinks about the image; iv) the polarity (positive/negative) of the answer, if applicable.

Downsampling and de-biasing. We noticed that the proportion of caption propositions was much larger than propositions deriving from other turns, which would cause a considerable imbalance towards facts that are always shared in the scoreboard. Therefore, we sampled 15% of the caption pairs (entailment and contradiction) on all datasets to make the distribution over manipulated turns be closer to uniform.

Furthermore, in preliminary experiments we observed that propositions could give away information on the true/false to A status. For instance, '*there is a zebra.*' can appear very often as an entailment (on the many photos showing zebras) but rarely as a contradiction (dialogues where Q spontaneously asks '*is there a zebra?*' and the answer is '*no*'). Besides, on rules that manipulate questions that are not polar (*what color is the dog? black.*), negation is always a contradiction. So the classifier could make predictions based on the lexical form alone. To counter this bias, we constructed a balanced training dataset by sampling from the original set while making sure that, for each p that A established to be true with respect to an image/dialogue, we also included an equal p paired with an image/dialogue in which it is established to be false. While this procedure reduced the size of the training set, we ensured that predictions on the true/false dimension would need to use the dialogue representations. We also limited the number of p of the same kind to 2,000 (1,000 as entailment, 1,000 as contradiction), to avoid having very common propositions like '*the photo is in color*' or '*it is sunny*' occurring too often.

Datasets used in the experiments. The following paragraphs discuss the final datasets used in the experiments (*i.e.* after downsampling captions and balancing the training set). The frequency over which turn was manipulated is shown in Figure 5. Although there is an imbalance towards later turns on the training set, the proportion of private/shared classes at turn 5 is relatively balanced (around 44.5/55.5), partially due to the fact that, at the last turn, no proposition is assigned a private class. Figure 6 shows the frequency of the number of turns that have been turned into propositions in a dialogue. Table 4 show the proportion of each type of proposition on the datasets. The training set has less propositions that do not derive from polar questions due to the balancing.

The propositions, paired to dialogue representations on each dialogue turn, with the class assigned to each tuple can be seen as a layer of annotation which is not predicted but constructed.

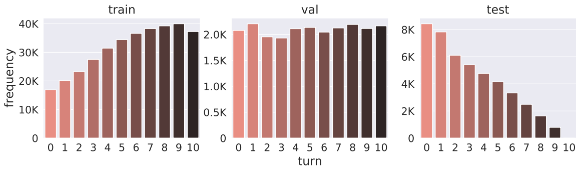


Figure 5: Distribution over manipulated turns. The test set has a different distribution because it has incomplete dialogues with varying length.

37.20% of the validation proposition types and 31.58% of the test proposition types appear among

[11] https://spacy.io/

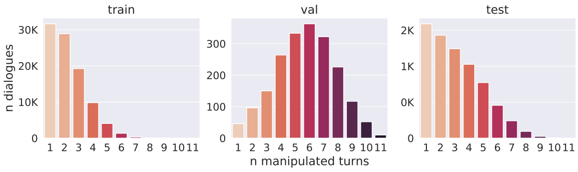


Figure 6: Number of manipulated turns per dialogue.

	train	valid	test
true to A	50.00	50.00	50.00
false to A	50.00	50.00	50.00
polar q, positive a	43.17	32.73	31.35
polar q, negative a	49.97	39.09	31.44
other q	6.84	28.16	37.19

Table 4: Proportion (%) of each type of proposition.

	train	valid	test
manipulation rule types	34	34	34
avr. manipulated turns per dialogue	2.28	5.72	3.13
min. propositions per dialogue	1	2	2
max. propositions per dialogue	16	26	22

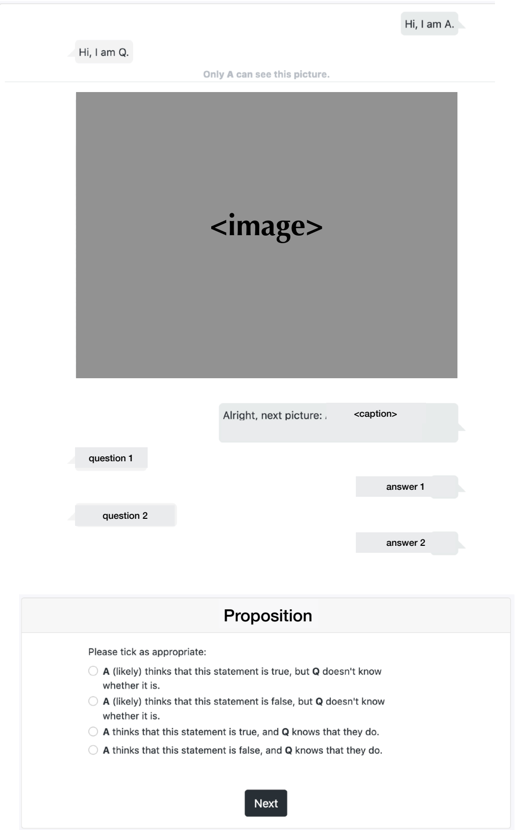
Table 5: Details of the proposition sets (after downsampling and balancing).

the training propositions. 82.68% of the validation propositions and 79.63% of the test propositions occur in only one dialogue. On average, a proposition appears in 12.77 dialogues in the training set, 1.91 dialogues in the validation set and 2.34 dialogues in the test set. 72.73% of the word types in the validation set and 63.00% of the word types in the test set occur in the training set.

Examples. Figure 10 shows dialogues from the training set and the propositions generated for each turn, after downsampling the caption propositions (but before balancing). Propositions can inherit grammatical or spelling problems from the dialogues themselves. Figure 1 in the main section contains all propositions, before downsampling.

Collecting dialogue representations. To collect the dialogue state representations, we adapted the original *train.py* and *evaluate.py* scripts.[12] To get the representation at turn 10 for A, we needed to feed a dummy next question made of the start and the end symbols with a question mark token in between.

Human Judgement. We randomly sampled 100 dialogues and one proposition on each of them.[13] Then we sampled a random turn up to which the corresponding dialogue would be shown. The annotators were non-native English speakers who worked as student assistants at the Computational Linguistics Lab of the University of Potsdam. The task was explained to the annotators verbally and then again in written form at the beginning of the annotation. All participants saw the same datapoints at a different random order, presented in a setting as shown in Figure 7, and had to select one of the four alternatives (which correspond to the main task TFxPS).


Hi, I am A.

Hi, I am Q.

Alright, next picture: <caption>

question 1

answer 1

question 2

answer 2

Proposition

Please tick as appropriate:

○ A (likely) thinks that this statement is true, but Q doesn't know whether it is.

○ A (likely) thinks that this statement is false, but Q doesn't know whether it is.

○ A thinks that this statement is true, and Q knows that they do.

○ A thinks that this statement is false, and Q knows that they do.

Figure 7: How the task was presented for the annotators.

B Reproducibility

In this section, we present further details of the implementation and additional results to support reproducibility. More information can also be found in the code documentation.

Hyperparameters. We used comet.ml's[14] implementation of the Bayes algorithm for hyperpa-

[12] https://github.com/vmurahari3/visdial-diversity

[13] 6 datapoints were later excluded due to a technical mismatch after refactoring.

[14] www.comet.ml

rameter search on A, main task, TFxPS, RL_DIV, aiming at maximizing accuracy on the validation set, as well as some manual selections. The (non-exhaustive) search space is shown in Table 6. The optimal configuration was then used in all experiments, with a maximum of 30 epochs and no early-stopping. A preliminary test with an even larger hidden dimension showed a very minor improvement. For each experiment, we used the configuration that led to the best performance on the validation set to get results on the test set. Each experiment took between 50 and 60 minutes.

The sentence encoder models listed on Table 6 are available at HuggingFace's Model Hub.[15]

Classifier architecture. The neural network was implemented using Pytorch 1.7.1. The proposition embeddings have 768 dimensions and the dialogue context embeddings have 512 dimensions. We used a sequential model from PyTorch with the following layers and dimensions:[16]

1. linear layer (in features=768+512, out features=1024, bias=True)
2. sigmoid function
3. dropout layer (p=0.1)
4. linear layer (in features=1024, out features=n labels in {2,3,4}, bias=True)
5. softmax function + cross entropy loss

The models have 1,315,844, 1,314,819 and 1,313,794 trainable parameters for the classification tasks with 4, 3 and 2 labels, respectively.

Infrastructure. The operating system used to run experiments was Linux, release 5.4.0-99-generic, processor x86_64. We had two GPUs available (NVIDIA GeForce GTX 1080 Ti), but each individual experiment used only one of them.

C Detailed Results

Table 7 shows the overall accuracy on all datapoints (comprising all turns in the test set). Table 8 and Table 9 show all results on the validation set.

On Figure 8 we split the accuracy per type of proposition. Propositions that derive from negative facts about the image (*'is there a dog? no.'*) seem to be harder than positive ones when they derive from earlier turns, but they are easier to correctly classify when they derive from later turns. Propositions deriving from questions that are not polar are harder (which may be a consequence of the balanced dataset selection that results in few propositions of this type for training). We also see that propositions derived from manipulating later turns are, in general, harder to classify.

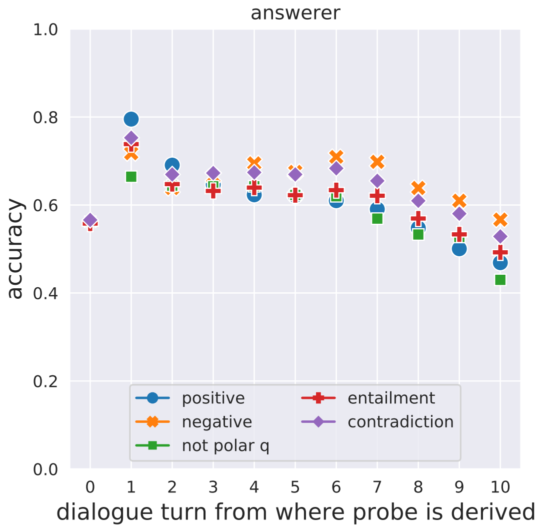


Figure 8: Accuracy per type of proposition (A, main, TFxPS, RL_DIV).

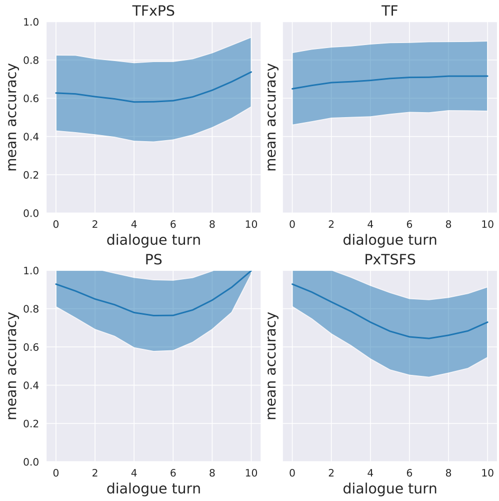


Figure 9: Mean accuracy on dialogue level over turns (A, main, TFxPS, RL_DIV).

When we consider each row of the scoreboard (representing the scoreboard at a given turn), we can inspect how accuracy evolves over turns, illustrated in Figure 9.

For the error analysis on captions, a right shift from private to shared means that the class at turn 0 is shared. Shifting only at the right turn means that it starts as shared and does not shift at any turn.

[15] https://huggingface.co/sentence-transformers

[16] During development, we also experimented with a shallow version, which did not perform very well, and a version with more layers, whose performance gain was not substantial.

Hyperparameter	Values	Selected
batch size	64, 128, 256, 512	512
clipping	0.0, 0.25, 0.5, 1, 5	1
dropout	0.0, 0.1, 0.3, 0.5	0.1
hidden dimension	64, 128, 256, 512, 1024	1024
learning rate	1e-5, 1e-3, 3e-5, 3e-3, 1e-2	0.001
random seed	2204, 10, 142, 54321	54321
sentence encoder	stsb-bert-base, paraphrase-mpnet-base-v2, nli-roberta-base-v2, stsb-roberta-base-v2	paraphrase-mpnet-base-v2

Table 6: Hyperparameters tried in the (non-exhaustive) search and selected hyperparameters used in all final experiments.

task		TFxPS			TF			PS			PxTSFS		
	model	(a)	(b)	(c)	(a)	(b)	(c)	(a)	(b)	(c)	(a)	(b)	(c)
A	main	62.04	62.33	61.78	71.02	70.92	70.79	80.94	81.24	80.79	73.06	73.36	73.47
	random r	35.10	35.56	35.12	52.48	51.82	53.17	60.35	60.65	60.46	47.95	48.65	48.62
	null r	37.66	37.52	37.71	50.61	50.60	50.61	60.25	60.24	60.21	50.64	50.86	50.62
Q	main	-	-	-	-	-	-	82.02	83.15	83.06	74.35	73.90	74.42
	random r	-	-	-	-	-	-	59.00	59.75	60.06	48.80	48.32	48.49
	null r	-	-	-	-	-	-	60.18	60.13	60.15	50.64	50.56	50.53

Table 7: Accuracy on the test set (all turns) for models (a) RL_DIV, (b) SL, (c) ICCV_RL.

task		TFxPS			TF			PS			PxTSFS		
	model	(a)	(b)	(c)	(a)	(b)	(c)	(a)	(b)	(c)	(a)	(b)	(c)
A	main	57.97	58.03	57.31	70.32	70.45	70.35	76.31	77.17	75.97	68.13	69.15	68.41
	random r	33.48	35.76	35.53	52.45	52.93	53.69	62.09	61.85	58.51	51.14	50.72	49.90
	null r	37.44	37.39	37.55	50.75	50.75	50.75	63.94	63.91	63.92	53.05	52.95	53.10
Q	main	-	-	-	-	-	-	78.49	79.74	79.22	71.62	71.37	71.28
	random r	-	-	-	-	-	-	62.30	60.80	61.16	52.12	51.58	51.69
	null r	-	-	-	-	-	-	63.89	63.82	63.86	53.17	52.98	52.99

Table 8: Accuracy on the validation set (turn 5) for models (a) RL_DIV, (b) SL, (c) ICCV_RL.

task		TFxPS			TF			PS			PxTSFS		
	model	(a)	(b)	(c)	(a)	(b)	(c)	(a)	(b)	(c)	(a)	(b)	(c)
A	main	62.46	62.55	62.30	69.59	69.83	69.52	85.00	85.34	84.82	74.74	75.13	74.97
	random r	33.52	33.86	33.54	52.42	52.85	53.55	59.54	59.64	59.88	49.53	49.55	50.02
	null r	34.84	34.75	34.88	50.74	50.74	50.74	59.75	59.73	59.71	51.14	51.01	51.13
Q	main	-	-	-	-	-	-	85.33	86.23	86.37	76.23	75.79	76.15
	random r	-	-	-	-	-	-	58.70	60.43	60.45	50.01	49.88	50.02
	null r	-	-	-	-	-	-	59.68	59.63	59.63	51.25	51.16	51.16

Table 9: Accuracy on the validation set (all turns) for models (a) RL_DIV, (b) SL, (c) ICCV_RL.

a dog that is looking at a herd of sheep.
none
are there any people? no.
there are no people.
there are people.
what color is the dog? whitish tan.
the dog is tan.
the dog is not tan.
is this in color? yes.
the image is in color.
the image is not in color.
is this a large field? very large.
none
is there tall grass? no.
there is no tall grass.
there is tall grass.
is it sunny? a little.
none
can you see a fence? no fences.
one cannot see any fence.
one can see a fence.
are there trees? 0.
there are no trees.
there are trees.
can you see mountains? i see a hillside.
none
any buildings? no buildings at all.
there are no buildings.
there are buildings.

a serving of dessert that includes various berries.
none
does this food look appetizing? no.
none
is veggies on dish? nope just fruit.
none
do you see apples? no apples.
one cannot see any apples.
one can see apples.
do you see grapes? no gapes at all.
one cannot see any grapes.
one can see grapes.
what is main fruit on dish? strawberries and blueberries.
none
do strawberries still have green on them? yes it does.
none
are blueberries large? no small and smashed.
the blueberries are not large.
the blueberries are large.
can you tell what color plate is? it is white bowl.
none
can you tell color of table? no,.
none
do you see people? no.
one cannot see any people.
one can see people.

this is a white kitchen with a window.
none
do you see a stove? yes.
one can see a stove.
one cannot see any stove.
what color is the stove? white and black.
the stove is white and black.
the stove is not white and black.
do you see a sink? yes.
one can see a sink.
one cannot see any sink.
can you see the fridge? no.
one cannot see any fridge.
one can see a fridge.
do the window have any curtains? no curtains.
the window do not have any curtains.
the window have any curtains.
do you see a dishwasher? no.
one cannot see any dishwasher.
one can see a dishwasher.
do you see any blinds? no blinds.
one cannot see any blinds.
one can see blinds.
any pictures on the wall? 0.
there are no pictures on the wall.
there are pictures on the wall.
do you see any people? no people are in the room.
one cannot see any people.
one can see people.
what color is the floors? grey.
the floors is grey.
the floors is not grey.

a black cat laying in the sun on a green bench.
one can see a black cat.
one cannot see a black cat.
is the bench chipped? no it's not.
the bench is not chipped.
the bench is chipped.
is it wood or metal? it looks metal to me.
none
is the cat sleep? no i see the eye to be open.
the cat is not sleep.
the cat is sleep.
any other cats? i can see only 1 cat.
none
any people? no.
there are no people.
there are people.
is it day? yes it is.
none
any sunshine? yes nice sunshine.
there is a sunshine.
there is no sunshine.
is this in a yard or park? it's a park.
none
is the field big? no in the picture.
the field is not big.
the field is big.
angry birds? i don't see any birds.
none

Figure 10: Example of generated propositions for VisDial dialogues (CC-BY 4.0) from the training set, after downsampling captions and before balancing.

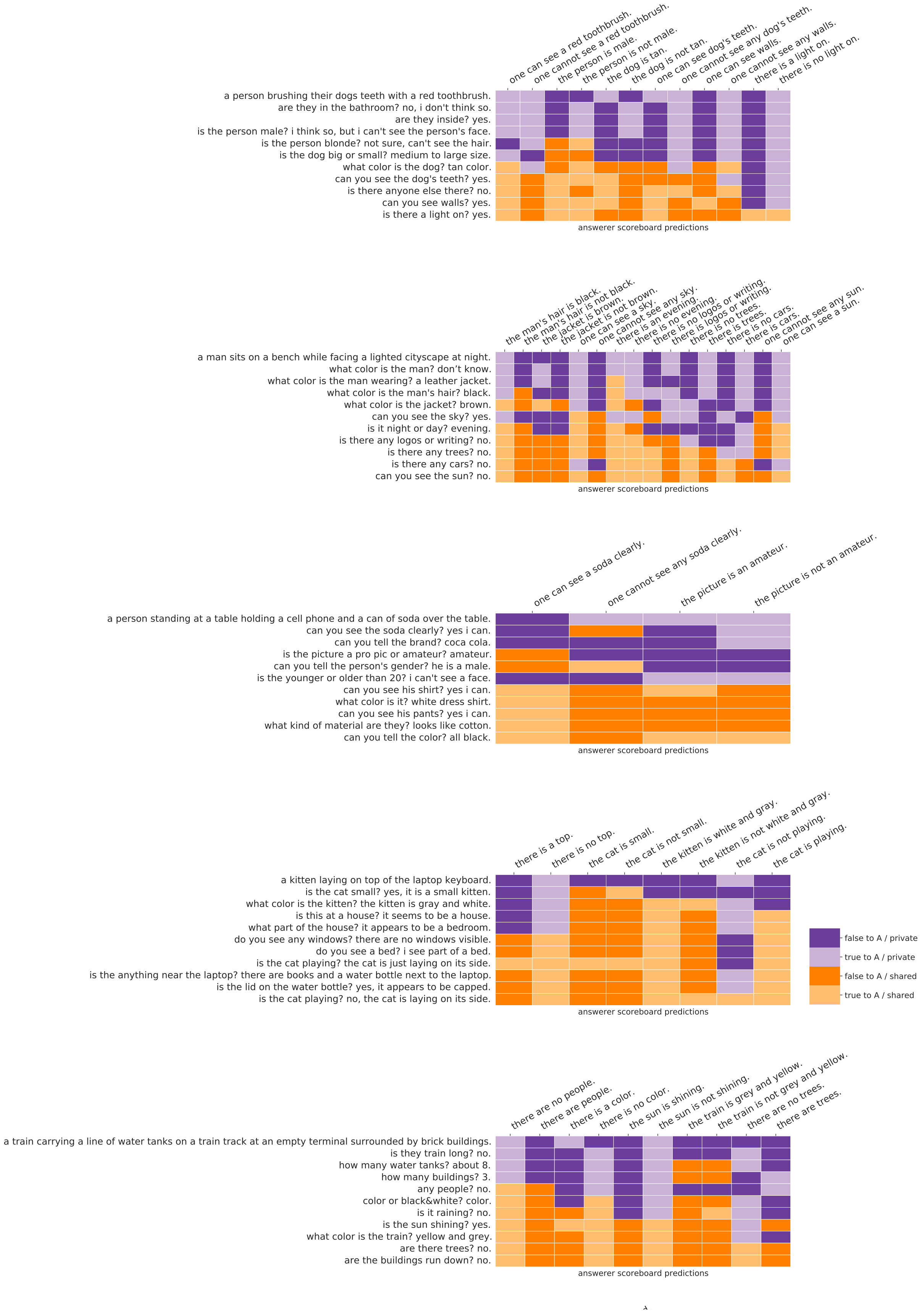


Figure 11: Examples of complete predicted scoreboards by A, main task, RL_DIV on TFxPS. All dialogues are from the VisDial validation set (CC-BY 4.0).

Focus on the Target's Vocabulary: Masked Label Smoothing for Machine Translation

Liang Chen, Runxin Xu, Baobao Chang[*]
Key Laboratory of Computational Linguistics, Peking University, MOE, China
leo.liang.chen@outlook.com
runxinxu@gmail.com chbb@pku.edu.cn

Abstract

Label smoothing and vocabulary sharing are two widely used techniques in neural machine translation models. However, we argue that simply applying both techniques can be conflicting and even leads to sub-optimal performance. When allocating smoothed probability, original label smoothing treats the source-side words that would never appear in the target language equally to the real target-side words, which could bias the translation model. To address this issue, we propose Masked Label Smoothing (MLS), a new mechanism that masks the soft label probability of source-side words to zero. Simple yet effective, MLS manages to better integrate label smoothing with vocabulary sharing. Our extensive experiments show that MLS consistently yields improvement over original label smoothing on different datasets, including bilingual and multilingual translation from both translation quality and model's calibration. Our code is released at PKUnlp-icler.

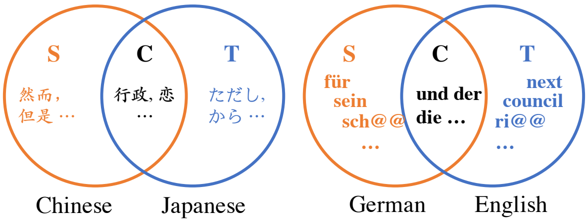


Figure 1: Venn diagram showing the structure of the shared vocabulary, which can be divided into three parts: Source (S), Common (C), and Target (T).

Model	**DE-EN**	**VI-EN**
Transformer	33.54	29.95
- w/ Label Smoothing (LS)	**34.76**	**30.73**
- w/ Vocabulary Sharing (VS)	33.83	29.36
- w/ LS+VS †	34.56	30.41

Table 1: Results in IWSLT'14 DE-EN and IWSLT'15 VI-EN datasets.† denotes consistent setting to Vaswani et al. (2017). Jointly adopting label smoothing and vocabulary sharing techniques cannot achieve further improvements, but leads to sub-optimal performance.

1 Introduction

Recent advances in Transformer-based (Vaswani et al., 2017) models have achieved remarkable success in Neural Machine Translation (NMT). For most NMT studies (Vaswani et al., 2017; Song et al., 2019; Lin et al., 2020; Liu et al., 2020; Ma et al., 2021), there are two widely used techniques to improve the quality of the translation: Label Smoothing (LS) and Vocabulary Sharing (VS). Label smoothing (Pereyra et al., 2017) turns the *hard* one-hot labels into a *soft* weighted mixture of the golden label and the uniform distribution over the whole vocabulary, which serves as an effective regularization technique to prevent over-fitting and over-confidence (Müller et al., 2019) of the model. In addition, vocabulary sharing (Xia et al., 2019) is another commonly used technique, which unifies the vocabulary of both source and target language into a whole vocabulary, and therefore the vocabulary is shared. It enhances the semantic correlation between the two languages and reduces the number of total parameters of the embedding matrices.

However, in this paper, we argue that jointly adopting both label smoothing and vocabulary sharing techniques can be conflicting, and leads to sub-optimal performance. Specifically, with vocabulary sharing, the shared vocabulary can be divided into three parts as shown in Figure 1. But with label smoothing, the soft label still considers the words at the source side that are impossible to appear at the target side. This would mislead the translation model and exerts a negative effect on the translation performance. As shown in Table 1, although introducing label smoothing or vocabulary sharing alone can improve the vanilla Transformer, jointly

[*] Corresponding author

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 665 - 671
May 22-27, 2022 ©2022 Association for Computational Linguistics

adopting both of them cannot obtain further improvements but achieves sub-optimal results.

To address the conflict of label smoothing and vocabulary sharing, we first propose a new mechanism named Weighted Label Smoothing (WLS) to control the smoothed probability distribution and its parameter-free version Masked Label Smoothing (MLS). Simple yet effective, MLS constrains the soft label not to assign soft probability to the words only belonging to the source side. In this way, we not only keeps the benefits of both label smoothing and vocabulary sharing, but also address the conflict of these two techniques to improve the quality of the translation.

According to our experiments, MLS leads to a better translation not only in scores like BLEU but also reports improvement in model's calibration. Compared with original label smoothing with vocabulary sharing, MLS outperforms in WMT'14 EN-DE(+0.47 BLEU), WMT'16 EN-RO (+0.33 BLEU) and other 7 language pairs including DE,RO-EN multilingual translation task.

2 Background

Label Smoothing The original label smoothing can be formalized as:

$$\hat{\boldsymbol{y}}^{LS} = \hat{\boldsymbol{y}}(1-\alpha) + \boldsymbol{\alpha}/K \quad (1)$$

K denotes the number of classes, α is the label smoothing parameter, $\boldsymbol{\alpha}/K$ is the soft label, $\hat{\boldsymbol{y}}$ is a vector where the correct label equals to 1 and others equal to zero and $\hat{\boldsymbol{y}}^{LS}$ is the modified targets.

Label smoothing is first introduced to image classification (Szegedy et al., 2016) task. Pereyra et al. (2017); Edunov et al. (2018) explore label smoothing's application in Sequence generation from token level and Norouzi et al. (2016) propose sentence level's label smoothing. Theoretically, Müller et al. (2019); Meister et al. (2020) all point out the relation between label smoothing and entropy regularization. Gao et al. (2020) explores the best recipe when applying label smoothing to machine translation. To generate more reliable soft labels, Lukasik et al. (2020) takes semantically similar n-grams overlap into consideration level label smoothing. Wang et al. (2020) proposes Graduate Label Smoothing that generate soft label according to the different confidence scores of model. To the best of our knowledge, we are the first to investigate label smoothing's influence on machine translation from the perspective of languages.

Category	DE->EN	RO->EN	VI->EN
Source	39%	50%	36%
Common	20%	8%	11%
Target	41%	42%	53%

Table 2: The distribution of different categories of the shared vocabulary forWMT'14 DE-EN, WMT'16 RO-EN, and IWSLT'15 VI-EN datasets. The proportion of tokens belonging to source category is up to 50%, which might mislead the translation model.

Vocabulary Sharing Vocabulary sharing is widely applied in most neural machine translation studies (Vaswani et al., 2017; Song et al., 2019; Lin et al., 2020). Researchers have conducted in-depth studies in Vocabulary Sharing. Liu et al. (2019) propose shared-private bilingual word embeddings, which give a closer relationship between the source and target embeddings. While Kim et al. (2019) point out that there is an vocabulary mismatch between parent and child languages in shared multilingual word embedding.

3 Conflict Between Label Smoothing and Vocabulary Sharing

Words or subwords in a language pair's joint dictionary can be categorized into three classes: **source**, **common** and **target** using Venn Diagram according to their belonging to certain language as depicted in Figure 1. This can be achieved by checking whether one token in the joint vocabulary also belongs to the source/target vocabulary. We formalized the categorization algorithm in Appendix A.

Then we compute the tokens' distribution in different translation directions as shown in Table 2. Tokens in source class account for a large proportion up to 50%. When label smoothing and vocabulary sharing are together applied, the smoothed probability will be allocated to words that belong to the source class. Those words have zero overlap with the possible target words, therefore they have no chance to appear in the target sentence. Allocating smoothed probability to them might introduce extra bias for the translation system during training process, unavoidably leading to a higher translation perplexity as also revealed by Müller et al. (2019).

Table 3 reveals the existence of conflict, that the joint use of label smoothing and vocabulary sharing doesn't compare with solely use one technique in all language pairs with a maximum loss of 0.32 BLEU score.

4 Methods

4.1 Weighted Label Smoothing

To deal with the conflict when executing label smoothing, we propose a plug-and-play Weighted Label Smoothing mechanism to control the smoothed probability's distribution.

Weighted Label Smoothing(WLS) has three parameters β_t, β_c, β_s apart from the label smoothing parameter α, where the ratio of the three parameters represents the portion of smoothed probability allocated to the target, common and source class and the sum of the three parameters is 1. The distribution within token class follows a uniform distribution. WLS can be formalized as:

$$\hat{\boldsymbol{y}}^{WLS} = \hat{\boldsymbol{y}}(1-\alpha) + \boldsymbol{\beta} \tag{2}$$

where $\hat{\boldsymbol{y}}$ is a vector where the element corresponding to the correct token equals to 1 and others equal to zero. $\boldsymbol{\beta}$ is a vector that controls the distribution of probability allocated to incorrect tokens. We use t_i, c_i, s_i to represent probability allocated to the i-th token in the target,common,source category, all of which form the distribution controlling vector $\boldsymbol{\beta}$ with $\sum_i^K \beta_i = \alpha$. The restriction can be formalized as:

$$\sum t_i : \sum c_i : \sum s_i = \beta_t : \beta_c : \beta_s \tag{3}$$

4.2 Masked Label Smoothing

Based on the Weight Label Smoothing mechanism, we can now implement Masked Label Smoothing by set β_s to 0 and regard the target and common category as one category. In this way, Masked Label Smoothing is parameter-free and implicitly injects external knowledge to the model. And we have found out that this simple setting can reach satisfactory results according our experiments.

We illustrate different label smoothing methods in Figure 2. It is worth noticing that MLS is different from setting WLS's parameters to 1-1-0 since there might be different number of tokens in the common and target vocab.

5 Experiments

5.1 Task Settings

For bilingual translation, we conduct experiments on 7 translation tasks. We choose language pairs that have different ratio of common subwords. These include WMT'14 DE-EN,EN-DE, IWSLT'14 DE-EN, IWSLT'15 VI-EN, WMT'16 RO-EN,EN-RO and CASIA ZH-EN.

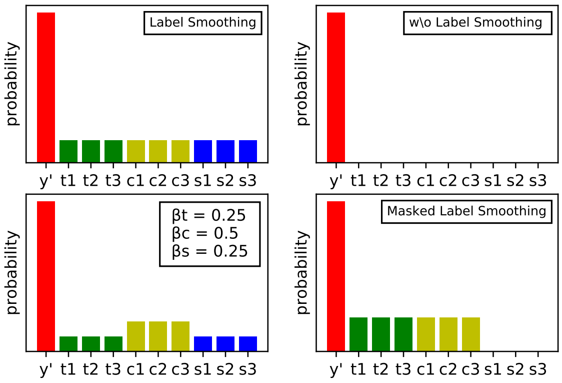


Figure 2: Illustration of different label smoothing methods. The height of each bar in the graph denoted the probability allocated to each token. y' is the current token during current decoding phase. We assume that there are only 10 tokens in the joint vocabulary and t1-t3 belongs to target class, c1-c3 belongs to common class and s1-s3 belongs to source class.

We use the official train-dev-test split of WMT'14, 16 and IWSLT'14, 15 datasets. For CASIA ZH-EN dataset, we randomly select 5000 sentences as development set and 5000 sentences as test set from the total dataset.

For multilingual translation, we combine the WMT'16 RO-EN and IWSLT'14 DE-EN datasets to formulate a RO,DE-EN translation task. We also make a balanced multilingual dataset that has equal numbers of DE-EN and RO-EN training examples to reduce the impact of imbalance languages and to explore how MLS performs under different data distribution condition in multilingual translation.

We apply the Transformer base (Vaswani et al., 2017) model as our baseline model. We fix the label smoothing parameter α to 0.1 in the main experiments and individually experiment and examine the performance of MLS under different α.

We use compound_split_bleu.sh from fairseq to compute the final bleu scores. The inference ECE score[1] and chrF score[2] are computed through open source scripts. We list the concrete training and evaluation settings in Appendix B.

5.2 Results

Bilingual Table 3 shows the results of bilingual translation experiments. The results reveal the conflict between LS and VS that models with only LS

[1] https://github.com/shuo-git/InfECE

[2] https://github.com/m-popovic/chrF

(a) Bilingual Translation

	WMT'16		**IWSLT'14**	**WMT'14**		**IWSLT'15**	**CASIA**
Model	RO-EN	EN-RO	DE-EN	DE-EN	EN-DE	VI-EN	ZH-EN
Transformer	22.03	19.61	33.54	30.85	27.21	29.95	20.66
- w/ VS	22.20	19.91	33.83	31.08	27.51	29.36	20.88
- w/ LS	22.96	20.68	34.76	31.14	27.53	**30.73**	21.10
- w/ LS+VS	22.89	20.59	34.56	30.98	27.44	30.41	21.04
- w/ MLS (ours)	**23.22****	**20.88****	**35.04****	**31.43***	**27.91***	30.57*	**21.23***

(b) Multilingual Translation

	IWSLT'14+WMT'16			**IWSLT'14+WMT'16†**		
Model	**DE,RO-EN**	**DE-EN**	**RO-EN**	**DE,RO-EN**	**DE-EN**	**RO-EN**
- w/ LS+VS	33.78	37.24	23.15	33.25	37.44	20.40
- w/ MLS (ours)	**34.10****	**37.53****	**23.19**	**33.53****	**37.77****	**20.86****

Table 3: Results of bilingual translation tasks (a) and multilingual translation (b). † denotes the balanced version of multilingual translation data. Same conflict between LS and VS occurs in all language pairs. Our MLS outperforms the original label smoothing with vocabulary sharing with significance levels when of p < 0.01 (**), p < 0.05 (*) and also beats individually using LS or VS in most cases.

surpass models with both LS and VS in all experiments. Our Masked Label Smoothing obtained consistent improvements over original LS+VS in all tested language pairs significantly.

The effectiveness of MLS maintained under different α value as shown in Table 4 for both BLEU and chrF scores. Similar to Gao et al. (2020)'s conclusion, we find that a higher α can generally improve the bilingual translation quality. And applying MLS can further improve the results. It shows that not only the probability increase in target vocabulary, but also the allocation of smoothed probabilities in different languages matters in the improvement of translation performance.

Multilingual As shown in Table 3, MLS achieves consistent improvement over the original label smoothing in both the original and the balanced multilingual translation dataset under all translation directions. In the original combined dataset, direction RO-EN (400K) has much more samples than DE-EN (160K). We do not apply a resampling strategy during training in order to investigate how the imbalance condition affects different models' performance. The balanced version cuts down samples in RO-EN direction to the same number as in DE-EN direction.

Compared with the imbalance version, the balanced version gave better BLEU scores in DE-EN direction while much worse performance in RO-EN translation for both the original label smoothing and MLS. It indicates that the cut down on RO-EN training examples does weaken the generalization of model in RO-EN translation however doesn't influence the DE-EN translation quality since the RO-EN data might introduce bias to the training process for DE-EN translation.

Even under imbalance condition, MLS can give a better performance (37.53) compared to original LS in the balance condition (37.44). It implies that MLS can relieve the imbalance data issue in multilingual translation. However, the improvement in relative high-resources direction (RO-EN) is not as significant as in the balanced condition. We guess that label smoothing has more complex influence on multilingual model due to the increase of languages and relation among different languages. We leave those questions for future exploration.

(a) EN-RO

Scores	BLEU(chrF)		
α	0.1	0.3	0.5
LS+VS	20.54(45.54)	20.65(45.79)	20.62(45.7)
MLS	**20.57(45.68)**	**20.99(46.29)**	**21.10(46.4)**

(b) RO-EN

Scores	BLEU(chrF)		
α	0.1	0.3	0.5
LS+VS	22.54(47.09)	22.95(47.29)	22.98(47.23)
MLS	**22.89(48.23)**	**23.10(48.36)**	**23.07(47.39)**

Table 4: Individual experiment on α. BLEU and chrF scores are reported under different label smoothing α on WMT'16 EN-RO (a) and RO-EN (b) datasets.

β_t	β_c	β_s	**RO-EN**	**EN-RO**	**DE-EN**
-	-	-	22.80	23.15	30.94
1/3	1/3	1/3	22.68	23.19	**31.40**
1/2	1/2	0	**23.05**	23.19	31.18
1/2	0	1/2	22.86	23.01	31.33
0	1/2	1/2	22.22	**23.33**	30.85
1/2	1/4	1/4	22.73	23.16	30.92

Table 5: Value "-" denotes the original label smoothing. WLS generally can improve the translation quality with appropriate parameters. Scores are computed using the development set of each direction.

6 Discussion

6.1 Exploring of Weighted Label Smoothing

As reported in Table 5, we explore the influence of different WLS on multiple tasks including WMT'16 RO-EN,EN-RO and WMT'14 DE-EN.

According to the result, though the best BLEU score's WLS setting vary from different tasks and there seems to exist a more complex relation between the probability allocation and the BLEU score, we still have two observations. First, applying WLS can generally boost the quality of translation compared to the original label smoothing. Second, only WLS with β_t, β_c, β_s each equals to 1/2-1/2-0 can outperform the original label smoothing on all tasks, which suggests the setting is the most robust one. Thus we recommend using this setting as the initial setting when applying WLS.

Furthermore, the most robust setting agrees with the form of MLS since they both allocate zero probability to the source category's tokens, which further proves the robustness of MLS.

6.2 Improvement in Model's Calibration and Translation Perplexity

Müller et al. (2019) have pointed out label smoothing prevents the model from becoming over-confident therefore improve the calibration of model. Since there is a training-inference discrepancy in NMT models, inference ECE score (Wang et al., 2020) better reflects models' real calibration.

To compute the ECE scores, we need to split the model's predictions into M bins according to the output confidence and calculate the weighted average of bin's confidence/accuracy difference as the ECE scores considering the number of samples in each bin.

Model	**DE-EN**	**VI-EN**	**DE,RO-EN**	**DE,RO-EN***
- w/ LS+VS	9.77	13.07	11.62	10.77
- w/ MLS	**9.67**	**12.63**	**11.37**	**8.82**

Table 6: Inference ECE score (less is better) on different translation tasks. $*$ denotes the balanced version of multilingual data. MLS leads to an average of 0.7 lower ECE score, suggesting better model calibration.

$$ECE = \sum_{i=1}^{M} \frac{|B_i|}{N} \left|\mathrm{acc}\left(B_i\right) - \mathrm{confidence}\left(B_i\right)\right|$$

where N is the number of total prediction samples and B_i is the number of samples in the i-th bin. acc (B_i) is the average accuracy in the i-th bin.

The score denotes the difference between accuracy and confidence of models' output during inference. Less ECE implies better calibration.

The inference ECE scores of our models are shown in Table 6. It turns out that models with MLS have lower Inference ECE scores on different datasets. The results indicate that MLS will lead to better model calibration.

We also find out that MLS leads to a significantly lower perplexity than LS during the early stage of training in all of our experiments. It's not surprising since zeroing the source side words' smoothed probability can decrease the perplexity. It can be another reason for model's better translation performance since it gives a better training initialization.

7 Conclusion

We reveal the conflict between label smoothing and vocabulary sharing techniques in NMT that jointly adopting the two techniques can lead to sub-optimal performance. To address this issue, we introduce Masked Label Smoothing to eliminate the conflict by reallocating the smoothed probabilities according to the languages' differences. Simple yet effective, MLS shows improvement over original label smoothing from both translation quality and model's calibration on a wide range of tasks.

8 Acknowledgements

We thank all reviewers for their valuable suggestions for this work. This paper is supported by the National Science Foundation of China under Grant No.61876004 and 61936012, the National Key Research and Development Program of China under Grant No. 2020AAA0106700.

9 Ethics Consideration

We collect our data from public datasets that permit academic use. The open-source tools we use for training and evaluation are freely accessible online without copyright conflicts.

References

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 355–364, New Orleans, Louisiana. Association for Computational Linguistics.

Yingbo Gao, Weiyue Wang, Christian Herold, Zijian Yang, and Hermann Ney. 2020. Towards a better understanding of label smoothing in neural machine translation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 212–223, Suzhou, China. Association for Computational Linguistics.

Yunsu Kim, Yingbo Gao, and Hermann Ney. 2019. Effective cross-lingual transfer of neural machine translation models without shared vocabularies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1246–1257, Florence, Italy. Association for Computational Linguistics.

Zehui Lin, Xiao Pan, Mingxuan Wang, Xipeng Qiu, Jiangtao Feng, Hao Zhou, and Lei Li. 2020. Pre-training multilingual neural machine translation by leveraging alignment information. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2649–2663, Online. Association for Computational Linguistics.

Xuebo Liu, Derek F. Wong, Yang Liu, Lidia S. Chao, Tong Xiao, and Jingbo Zhu. 2019. Shared-private bilingual word embeddings for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3613–3622, Florence, Italy. Association for Computational Linguistics.

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

Michal Lukasik, Himanshu Jain, Aditya Menon, Seungyeon Kim, Srinadh Bhojanapalli, Felix Yu, and Sanjiv Kumar. 2020. Semantic label smoothing for sequence to sequence problems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4992–4998, Online. Association for Computational Linguistics.

Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, Alexandre Muzio, Saksham Singhal, Hany Hassan Awadalla, Xia Song, and Furu Wei. 2021. Deltalm: Encoder-decoder pre-training for language generation and translation by augmenting pretrained multilingual encoders. *CoRR*, abs/2106.13736.

Clara Meister, Elizabeth Salesky, and Ryan Cotterell. 2020. Generalized entropy regularization or: There's nothing special about label smoothing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6870–6886, Online. Association for Computational Linguistics.

Rafael Müller, Simon Kornblith, and Geoffrey Hinton. 2019. *When Does Label Smoothing Help?* Curran Associates Inc., Red Hook, NY, USA.

Mohammad Norouzi, Samy Bengio, zhifeng Chen, Navdeep Jaitly, Mike Schuster, Yonghui Wu, and Dale Schuurmans. 2016. Reward augmented maximum likelihood for neural structured prediction. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Gabriel Pereyra, G. Tucker, J. Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *ICML*.

Christian Szegedy, V. Vanhoucke, S. Ioffe, Jonathon Shlens, and Z. Wojna. 2016. Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Shuo Wang, Zhaopeng Tu, Shuming Shi, and Yang Liu. 2020. On the inference calibration of neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3070–3079, Online. Association for Computational Linguistics.

Yingce Xia, Tianyu He, Xu Tan, Fei Tian, Di He, and Tao Qin. 2019. Tied transformers: Neural machine translation with shared encoder and decoder. In *AAAI*.

A Algorithm

Algorithm 1 Divide Token Categories

Input: List: S, T, J

Output: List: A,B,C

Description: S is the vocabulary list for source language, T for target language, J for joint vocabulary. A is the output vocabulary for source tokens, B for common tokens, C for target tokens.

```
Initialize empty list A,B,C
for i in J do
    if i in S and i in T then
        B.add(i)
    else
        if i in S then
            A.add(i)
        else
            C.add(i)
return A,B,C
```

B Experiment Details

We evaluate our method upon Transformer-Base (Vaswani et al., 2017) and conduct experiments under same hyper-parameters for fair comparison. We use fairseq (Ott et al., 2019) as the main code base.

Before training, we first apply BPE (Sennrich et al., 2016) to tokenize the corpus for 16k steps each language and then learn a joint dictionary. During training, the label smoothing parameter α is set to 0.1 except for Table 4's exploration in alpha values. We use Adam optimizer with betas to be (0.9,0.98) and learning rate is 0.0007. During warming up steps, the initial learning rate is 1e-7 and there are 1000 warm-up steps. We use a batch-size of 2048 together with an update-freq of 4 on two NVIDIA 3090 GPUs. Dropout rate is set to 0.3 and weight decay is set to 0.0001 for all experiments. We average the last 3 checkpoints to generate the final model in the main bilingual experiments before inferring on the test set. We use beam size as 5 during all testing.

Contrastive Learning-Enhanced Nearest Neighbor Mechanism for Multi-Label Text Classification

Xi'ao Su, Ran Wang, Xinyu Dai[*]
National Key Laboratory for Novel Software Technology, Nanjing University, China
Collaborative Innovation Center of Novel Software Technology and Industrialization, China
{nlp_suxa,wangr}@smail.nju.edu.cn, daixinyu@nju.edu.cn

Abstract

Multi-Label Text Classification (MLTC) is a fundamental and challenging task in natural language processing. Previous studies mainly focus on learning text representation and modeling label correlation. However, they neglect the rich knowledge from the existing similar instances when predicting labels of a specific text. To address this oversight, we propose a k nearest neighbor (kNN) mechanism which retrieves several neighbor instances and interpolates the model output with their labels. Moreover, we design a multi-label contrastive learning objective that makes the model aware of the kNN classification process and improves the quality of the retrieved neighbors during inference. Extensive experiments show that our method can bring consistent and considerable performance improvement to multiple MLTC models including the state-of-the-art pretrained and non-pretrained ones.

1 Introduction

Multi-Label Text Classification (MLTC) is a fundamental task in natural language processing, which can be found in many real-world scenarios such as web page tagging (Jain et al., 2016), topic recognition (Yang et al., 2016), sentiment analysis (Wang et al., 2016) and so on. Different from multi-class classification where only one label is identified as positive, MLTC aims to assign multiple labels from a predefined set to each text.

Till now, extensive research has been carried out to solve the MLTC task. Among them, some methods focus on learning enhanced text representation with deep neural networks (Kurata et al., 2016; Liu et al., 2016) or the label-wise attention mechanism (Xiao et al., 2019; Ma et al., 2021). Meanwhile, others try to model the label correlation by the sequential prediction (Nam et al., 2017; Yang et al., 2018), iterative reasoning (Wang et al., 2021), or graph neural networks (Ma et al., 2021).

Text	Labels
The **mutual information** of two random variables is commonly used in learning bayesian nets as well as in other fields ...	**math.ST** **math.IT** **stat.TH** **cs.IT** **cs.AI**
Mutual information is widely used, to measure the stochastic dependence of categorical random variables in order to address questions ...	**math.ST** **math.IT** **stat.TH** **cs.IT** **cs.AI** cs.LG

Table 1: An example of two papers from arXiv.

However, during inference, these methods neglect the rich knowledge which can be directly obtained from the existing training instances. Utilizing this knowledge can assist the model to predict more accurately. For example, Tab. 1 lists two papers from arXiv[1] along with their tags. Both papers research on "Mutual Information" and they have almost the same labels. If we are tagging the second paper, then we can easily get a good reference from the first one. Therefore, when predicting labels for a specific text, the model can get immediate and reliable help from the instances with similar texts.

To this end, for the first time, we solve the MLTC task by the use of k nearest neighbor (kNN) mechanism which can effectively utilize the knowledge from existing multi-label instances. Specifically, it retrieves several neighbor instances based on text representations generated by the MLTC model and interpolates the model prediction with their labels. Moreover, to make the model aware of the kNN process and improve the quality of retrieved neighbors, we propose to train the model with a contrastive learning (CL) objective. Existing super-

[*] Corresponding author.

[1] https://arxiv.org/

vised contrastive learning methods (Gunel et al., 2021; Li et al., 2021) are proposed under the conventional multi-class setting, where two instances are either positive or negative for each other. However, in MLTC, two instances may share some common labels while there may also be some labels that are unique to each instance. How to handle these cases is the key to utilizing contrastive learning in MLTC. We argue that simply treating these instance pairs as positive ones is sub-optimal due to the variable similarities in different instance pairs, which is verified in Section 4.2. To model more fine-grained correlations between multi-label instances, we design a multi-label contrastive learning objective with a dynamic coefficient for each instance pair based on the label similarity. Training with this objective encourages the model to generate closer representations for instance pairs with more shared labels and push away those pairs that have completely different labels. As a result, the kNN mechanism will retrieve instances that contain more relevant labels, thereby further improving the classification performance. It's worth noting that our method is of high versatility and can be directly applied to most existing MLTC models.

In summary, our contributions are as follows:

- We propose a k nearest neighbor mechanism for MLTC that directly utilizes the knowledge from the existing instances during inference.
- We design a multi-label contrastive learning objective which can effectively enhance the kNN mechanism for MLTC.
- Extensive experiments show that our method can consistently and considerably improve the performance of multiple existing MLTC models including the state-of-the-art pretrained and non-pretrained ones.

2 Related Work

Multi-label Text Classification Existing methods for MLTC mainly focus on learning text representation and modeling label correlation. At first, CNN (Kim, 2014; Kurata et al., 2016) and RNN-based (Liu et al., 2016) models were used to capture local and long-distance text dependencies. Besides, Xiao et al. (2019) proposed a label-specific attention network to focus on different tokens when predicting each label. The sequence generation model (Yang et al., 2018) and iterative reasoning mechanism (Wang et al., 2021) were utilized to model the label correlation. Furthermore, Ma et al. (2021) adopted graph neural networks based on label graphs. However, these methods are unable to refer to the existing instances that can guide the model to make better predictions.

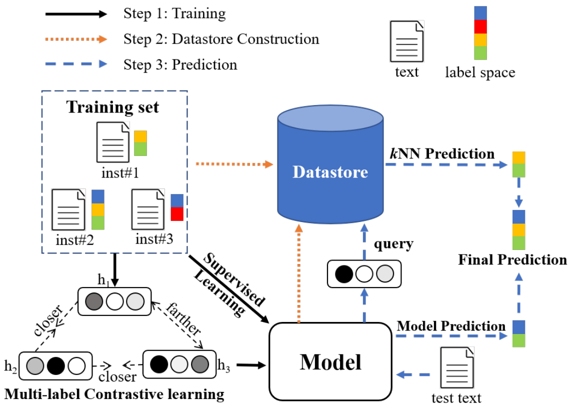


Figure 1: The overview of our proposed method.

Nearest Neighbor Methods in NLP Nearest neighbor methods have achieved great success in many NLP tasks such as language modeling (Khandelwal et al., 2020) and machine translation (Khandelwal et al., 2021; Zheng et al., 2021; Lin et al., 2021; Su et al., 2015). These methods utilize kNN retrieval in the inference stage based on context representation vectors which are generated by a converged model. Zheng et al. (2021) pointed out that simple application of the kNN method tends to introduce noise and we also found this issue in MLTC. Therefore, we design a multi-label contrastive learning objective to improve the quality of the retrieved neighbors. [2]

3 Proposed Method

In this section, we introduce our proposed method in detail. As depicted in Fig. 1, we design a k nearest neighbor mechanism for MLTC (Step 2, 3) and enhance it by training the model with a multi-label contrastive learning objective (Step 1).

3.1 Problem Formulation

Let $D = \{(x_i, y_i)\}_{i=1}^{N}$ be the MLTC training set consisting of N instances. Each x_i is a text and

[2] Contemporary with our work, KNN-BERT (Li et al., 2021) uses kNN and CL to enhance pretrained models' performance on multi-class classification. However, the way it uses kNN and sets positive/negative pairs in CL is inapplicable to multi-label scenarios due to its neglect of multiple non-exclusive labels in each instance, which is addressed by us in Section 3.3.

$y_i \in \{0,1\}^L$ denotes the corresponding multi-hot label vector where L is the total number of labels. The target of MLTC is to learn the mapping from the input text to the relevant labels.

3.2 Nearest Neighbor MLTC

To obtain knowledge from existing instances during inference, we propose a k nearest neighbor mechanism for MLTC including two steps: constructing a datastore of training instances (Step 2) and making the kNN prediction based on it (Step 3).

Datastore Construction Given an instance from the training set $(x_i, y_i) \in D$, the text representation vector $h_i = f(x_i)$ can be generated by an MLTC model. Then the multidimensional datastore D' can be constructed offline by a single forward pass over each training instance: $D' = \{(h_i, y_i)\}_{i=1}^N$.

Prediction In the inference stage, given an input text x, the model outputs the prediction vector $\hat{y}_{\text{Mo}} \in \{p|p \in [0,1]\}^L$. The model also outputs the text representation $f(x)$, which is utilized to query the datastore D' according to the euclidean distance to obtain the k nearest neighbors: $\mathcal{N} = \{(h_i, y_i)\}_{i=1}^k$. Then the kNN prediction can be made by:

$$\hat{y}_{\text{kNN}} = \sum_{i=1}^{k} \alpha_i y_i,\ \alpha_i = \frac{e^{-d(h_i, f(x))/\tau}}{\sum_j e^{-d(h_j, f(x))/\tau}} \quad (1)$$

where $d(\cdot,\cdot)$ indicates the euclidean distance, τ is the kNN temperature, and α_i denotes the weight of the i-th neighbor. Intuitively, the closer a neighbor is to the test instance, the larger its weight is. The final prediction is calculated as the combination of the base model output and the kNN prediction: $\hat{y} = \lambda \hat{y}_{\text{kNN}} + (1-\lambda)\hat{y}_{\text{Mo}}$ where λ is the proportion parameter.

3.3 Multi-Label Contrastive Learning

In MLTC, a model is usually trained by supervised learning with the binary cross-entropy (BCE) loss which is unaware of the kNN retrieval process. In consequence, retrieved neighbors may not have similar labels to the test instance and provide little help for the prediction. To fill this gap, we propose to train the model with a multi-label contrastive learning objective.

Existing supervised contrastive learning methods tried to narrow distances between instances from the same class and push away those from different classes. However, in MLTC, two instances may share some common labels while there may also be some labels that are unique to each instance. How to handle these cases is the key to utilizing contrastive learning in MLTC. Therefore, to model complex correlations among the multi-label instances, we design a dynamic coefficient based on the label similarity.

Considering a data minibatch of size b, we define a function to output all the other instances for a specific instance i: $g(i) = \{k|k \in \{1, 2, \cdots, b\}, k \neq i\}$. The contrastive loss for each instance pair (i, j) can be calculated as:

$$\mathcal{L}_{\text{con}}^{ij} = -\beta_{ij} \log \frac{e^{-d(z_i, z_j)/\tau'}}{\sum_{k \in g(i)} e^{-d(z_i, z_k)/\tau'}} \quad (2)$$

$$C_{ij} = y_i^\top \cdot y_j,\ \beta_{ij} = \frac{C_{ij}}{\sum_{k \in g(i)} C_{ik}} \quad (3)$$

where $d(\cdot,\cdot)$ is the euclidean distance, τ' is the contrastive learning temperature and $z_i = f(x_i)$ denotes the text representation. C_{ij} denotes the label similarity between i, j which is computed by the dot product of their label vectors. The dynamic coefficient β_{ij} is the normalization of C_{ij}.

The contrastive loss for the whole minibatch is the summation over all the instance pairs: $\mathcal{L}_{\text{con}} = \sum_i \sum_{j \in g(i)} \mathcal{L}_{\text{con}}^{ij}$. For a pair of instances (i, j), the greater label similarity C_{ij} will bring larger coefficient β_{ij}, thereby increasing the value of their loss term $\mathcal{L}_{\text{con}}^{ij}$. As a result, their distance $d(z_i, z_j)$ will be optimized to be closer. Meanwhile, if they have no shared labels ($\beta_{ij} = C_{ij} = 0$), then the value of $\mathcal{L}_{\text{con}}^{ij}$ is also zero and their distance $d(z_i, z_j)$ will only appear in the denominators of other terms. Consequently, their distance will have negative gradients and be optimized to become far.

Denoting BCE loss as $\mathcal{L}_{\text{BCE}}$, the overall training loss of our method is: $\mathcal{L} = \mathcal{L}_{\text{BCE}} + \gamma \mathcal{L}_{\text{con}}$. The parameter γ controls the trade-off between losses.

Dataset	I	L	$\overline{\mathbf{L}}$	$\overline{\mathbf{W}}$
AAPD	55,840	54	2.4	163
RCV1-V2	804,414	103	3.2	124

Table 2: Statistics of the datasets. **I** and **L** denote the total number of instances and labels. $\overline{\mathbf{L}}$ and $\overline{\mathbf{W}}$ denote the average number of labels and words per instance.

4 Experiments

In this section, we conduct multiple experiments to evaluate the efficacy of our method. Implementation details and the overhead of our method can be found in Appendix A and B respectively.

4.1 Settings

Datasets To evaluate our method, we conduct experiments on two benchmark datasets AAPD (Yang et al., 2018) and RCV1-V2 (Lewis et al., 2004). The dataset statistics are listed in Tab. 2.

Evaluation Metrics Following the previous work (Yang et al., 2018), we adopt hamming loss and micro-F1 score as our evaluation metrics.

Baseline We adopt the following models as our baselines and apply our method to all of them:

CNN (Kim, 2014) uses multiple convolutional kernels to extract local text representations.

LDGN (Ma et al., 2021) is the state-of-the-art non-pretrained MLTC model. It is based on the label-wise attention network and a GCN.

BERT (Devlin et al., 2019) is a Transformer-based pretrained language model. Its [CLS] representation is used to do the classification.[3]

Models	AAPD		RCV1-V2	
	HL(-)	F1(+)	HL(-)	F1(+)
CNN	0.02378	69.60	0.00946	83.76
+ours	0.02248	71.69	0.00824	86.14
LDGN	0.02478	70.59	0.00863	86.00
+ours	0.02296	71.38	0.00768	87.29
BERT	0.02257	74.03	0.00766	87.54
+ours	**0.02167**	**75.18**	**0.00715**	**88.36**

Table 3: Performance of all the models. HL and F1 denote the hamming loss and micro-F1 (%). The symbol '+'/'-' indicates that the higher/lower the value is, the better the model performs. Best results are marked bold.

4.2 Results

Main Experiments As shown in Tab. 3, our method can bring consistent and considerable performance improvements to all of the models. For example, our method has improved the micro-F1 of CNN by 2.09% on AAPD and 2.38% on RCV1-V2 respectively. Moreover, both the state-of-the-art LDGN and powerful BERT can still benefit a lot from our method. Specifically, when equipped with our method, the non-pretrained model LDGN obtains competitive performances compared to the pretrained model BERT on the larger RCV1-V2.

Models	AAPD	RCV1-V2
CNN	69.60	83.76
CNN+kNN	70.19	85.21
CNN+CL	69.43	83.84
CNN+CL+kNN	71.69	86.14
LDGN	70.59	86.00
LDGN+kNN	70.73	86.76
LDGN+CL	70.44	86.51
LDGN+CL+kNN	71.38	87.29
BERT	74.03	87.54
BERT+kNN	74.22	87.84
BERT+CL	73.85	87.74
BERT+CL+kNN	75.18	88.36

Table 4: Micro-F1 (%) of the ablation tests. kNN and CL denote the k nearest neighbor mechanism and contrastive learning objective respectively.

Ablation Test As mentioned above, our method consists of a k nearest neighbor mechanism (denoted as kNN) and a multi-label contrastive learning objective (denoted as CL). We demonstrate the effect of each component via an ablation test.

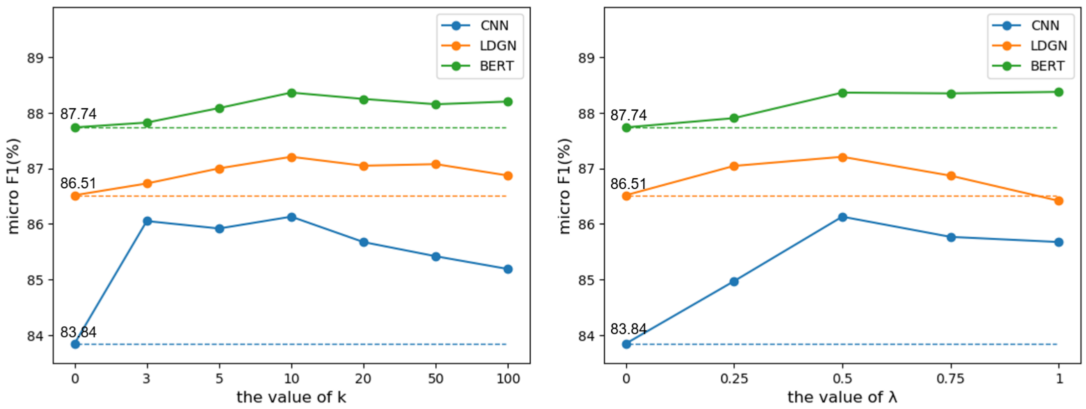
As shown in Tab. 4, the kNN mechanism can consistently improve the performance of the base models. Moreover, when equipped with our contrastive learning loss, although performances of the base models remain consistent, the improvements brought by the kNN mechanism have increased by a large margin. This verifies that our CL objective does effectively enhance the kNN mechanism.

Models	AAPD		RCV1-V2	
	w/o β	w/ β	w/o β	w/ β
CNN	71.19	71.69	85.27	86.14
LDGN	71.06	71.38	86.78	87.29
BERT	74.66	75.18	88.08	88.36

Table 5: Micro-F1 (%) of our methods with or without the dynamic coefficient β.

Analysis of Dynamic Coefficient In existing CL methods, two instances are either positive or negative for each other. To model more fine-grained similarity between instances, we proposed a dynamic coefficient β for each CL loss term (see Eq. 2,3).

[3] We also experimented on RoBERTa but it was outperformed by BERT in our task. Therefore, we choose BERT as the baseline pretrained model in our experiments.



(a) Anaylsis of the number of neighbors

(b) Analysis of the proportion of kNN prediction

Figure 2: Hyperparameter analysis of the kNN mechanism on the RCV1-V2 dataset.

To verify the necessity of β, we also apply the simple extension of existing CL methods to MLTC[4]. As shown in Tab. 5, our method outperforms the simple extension method in all cases, which verifies the necessity of considering the fine-grained similarity between multi-label instances.

Analysis of kNN Paramters Here we conduct a parameter analysis of our kNN mechanism on the RCV1-V2 dataset. As shown in Fig. 2(a), for all the models, the performance improves at first and then decreases as the k increases. Moreover, when referring to neighbor instances ($k > 0$), the performance is always better than only using the model output ($k = 0$), which verifies the necessity of utilizing the knowledge from the existing instances. Fig. 2(b) demonstrates the trend of model performance with λ. In general, the trend is similar to that of k which further confirms that only using the model prediction ($\lambda = 0$) is sub-optimal. It's worth noting that on the BERT model, completely using neighbors' prediction ($\lambda = 1$) is highly competitive compared to the uniform combination ($\lambda = 0.5$) which performs the best on the other base models.

Impact of Contrastive Learning To further analyze the impact of our contrastive learning objective, for each test instance, we count the average proportion of shared labels to all labels brought by its nearest neighbors. As shown in Tab. 6, after training the model with contrastive learning, the retrieved instances contain more shared labels with the test instance, which further proves that CL does improve the quality of the retrieved neighbors. An intuitive example can be found in Appendix C.

Models	AAPD		RCV1-V2	
	w/o CL	w/ CL	w/o CL	w/ CL
CNN	64.5	65.5	82.7	84.2
LDGN	63.1	64.2	84.4	84.9
BERT	67.8	68.5	85.5	86.4

Table 6: The average proportion (%) of the shared labels to all labels brought by the nearest neighbors to each test instance with or without our CL objective.

5 Conclusion

In this paper, we proposed a k nearest neighbor mechanism along with a multi-label contrastive learning objective for MLTC. Extensive experiments verified the effectiveness of our method and revealed the source of performance improvements our method brings. For future work, we will explore how to improve the performance of MLTC models directly with contrastive learning.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive comments. This work was supported by NSFC Projects (Nos. 61936012 and 61976114), the National Key RD Program of China (No. 2018YFB1005102).

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of

[4] The extension method can be obtained by setting all the C_{ij} greater than 1 to 1 in Eq. 3. This means if two instances have any shared label, they are considered to be a positive pair.

deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Himanshu Jain, Yashoteja Prabhu, and Manik Varma. 2016. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 935–944. ACM.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Gakuto Kurata, Bing Xiang, and Bowen Zhou. 2016. Improved neural network-based multi-label classification with better initialization leveraging label cooccurrence. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 521–526. The Association for Computational Linguistics.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5:361–397.

Linyang Li, Demin Song, Ruotian Ma, Xipeng Qiu, and Xuanjing Huang. 2021. KNN-BERT: fine-tuning pre-trained models with KNN classifier. *CoRR*, abs/2110.02523.

Huan Lin, Liang Yao, Baosong Yang, Dayiheng Liu, Haibo Zhang, Weihua Luo, Degen Huang, and Jinsong Su. 2021. Towards user-driven neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4008–4018. Association for Computational Linguistics.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2873–2879. IJCAI/AAAI Press.

Qianwen Ma, Chunyuan Yuan, Wei Zhou, and Songlin Hu. 2021. Label-specific dual graph neural network for multi-label text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3855–3864. Association for Computational Linguistics.

Jinseok Nam, Eneldo Loza Mencía, Hyunwoo J. Kim, and Johannes Fürnkranz. 2017. Maximizing subset accuracy with recurrent neural networks in multi-label classification. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5413–5423.

Jinsong Su, Deyi Xiong, Biao Zhang, Yang Liu, Junfeng Yao, and Min Zhang. 2015. Bilingual correspondence recursive autoencoder for statistical machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1248–1258. The Association for Computational Linguistics.

Ran Wang, Robert Ridley, Xi'ao Su, Weiguang Qu, and Xinyu Dai. 2021. A novel reasoning mechanism for multi-label text classification. *Inf. Process. Manag.*, 58(2):102441.

Yaqi Wang, Shi Feng, Daling Wang, Ge Yu, and Yifei Zhang. 2016. Multi-label chinese microblog emotion classification via convolutional neural network. In *Web Technologies and Applications - 18th Asia-Pacific Web Conference, APWeb 2016, Suzhou, China, September 23-25, 2016. Proceedings, Part I*, volume 9931 of *Lecture Notes in Computer Science*, pages 567–580. Springer.

Lin Xiao, Xin Huang, Boli Chen, and Liping Jing. 2019. Label-specific document representation for multi-label text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International*

Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 466–475. Association for Computational Linguistics.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 3915–3926. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1480–1489. The Association for Computational Linguistics.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. 2021. Adaptive nearest neighbor machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 368–374. Association for Computational Linguistics.

A Implementation Details

We implement all the methods relying on the PyTorch library[5]. We also use Faiss (Johnson et al., 2021) for fast nearest neighbor search. For CNN and BERT, we directly use the representations from the last hidden layer to construct the datastore. As for the LDGN which generates label-specific text representations, we perform a max-pooling operation on all the l vectors to get the single representation vector.

We train all the models on both datasets up to 30 epochs with an early stop of 3 patience and use the Adam optimizer with a learning rate of 1×10^{-3}. For all the models on AAPD, we use a batch size of 128. On RCV1-V2, we use a batch size of 512 for CNN and LDGN, and 128 for BERT due to its huge memory usage. As for the hyperparameters of our proposed method, $\lambda = 0.5,\ \tau = 1,\ \tau' = 10$ are adopted for all the cases. Besides, we use $k = 5,\ \gamma = 0.1$ for all the models on AAPD and $k = 10,\ \gamma = 0.01$ for those on RCV1-V2.

Models	AAPD	RCV1-V2
CNN	0.09 GB	1.46 GB
LDGN	0.11 GB	1.84 GB
BERT	0.17 GB	2.60 GB

Table 7: Disk usage of each datastore.

Models	AAPD		RCV1-V2	
	w/o kNN	w/ kNN	w/o kNN	w/ kNN
CNN	3.18	3.25	2.89	6.17
LDGN	5.47	7.29	7.61	9.67
BERT	264.89	267.57	265.96	270.73

Table 8: Inference time (ms/text) of different models with or without the kNN prediction. All results are tested with an RTX-2080Ti GPU.

B Space and Time Overhead

In the training stage, the overhead of contrastive learning is negligible compared to supervised learning, so we do not report it here. Most of the overhead lies in the kNN classifier. The disk usage of each datastore is shown in Tab. 7. The inference time per text of different models with or without the kNN prediction on each dataset is listed in

[5] https://pytorch.org/

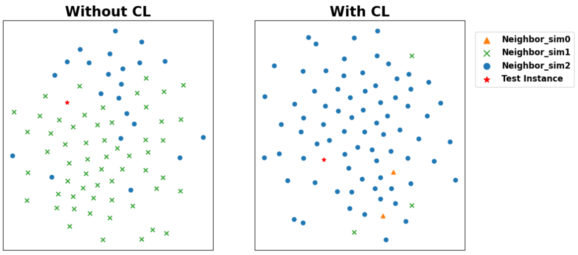


Figure 3: TSNE visualization where the red star stands for the test instance. Neighbors with different similarities to the test instance are plotted with different marks.

Tab. 8. It's worth noting that the extra inference time brought by our method does not exceed 5ms in all cases.

C Case Study: TSNE Visualization

In Fig. 3, we use the TSNE visualization tool to plot the CNN representations of a test instance and its 80 nearest neighbors with or without our CL objective. We use different marks to plot neighbors with different label similarities (C_{ij} in Eq. 3) to the test instance. As demonstrated in the left part, without contrastive learning, most of the nearest neighbors have only the similarity of 1 (green crosses). However, in the right part, with our CL objective, the test instance is surrounded by neighbors which have a high label similarity of 2 (blue circles). This confirms that our CL objective does improve the quality of the retrieved neighbors.

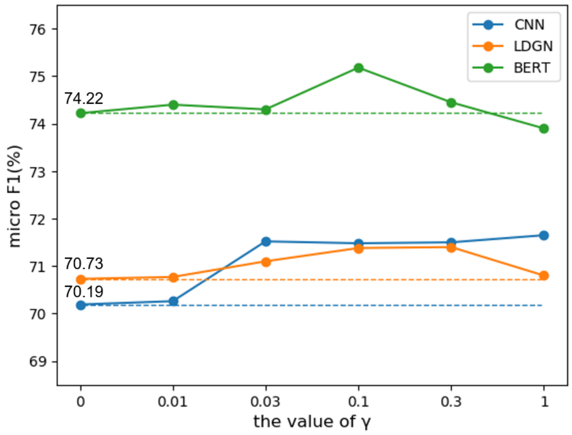


Figure 4: Analysis of the proportion of our contrastive learning objective based on each model.

D Analyzing the Proportion of Contrastive Learning

In this section, we analyze how the proportion of contrastive learning affects the performance of our method. As shown in Fig. 4, when trained with the contrastive learning objective ($\gamma > 0$), the performance of our method is better than that without contrastive learning ($\gamma = 0$) in most cases. However, when training the BERT model, too high proportion of contrastive learning ($\gamma = 1$) even hurts the performance. Besides, different base models have the different γ values for their optimal performance, which indicates that the proportion of contrastive learning to the overall training objective is crucial to the performance and varies with different model structures.

NoisyTune: A Little Noise Can Help You Finetune Pretrained Language Models Better

Chuhan Wu[†] **Fangzhao Wu**[‡*] **Tao Qi**[†] **Yongfeng Huang**[†]
[†]Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
[‡]Microsoft Research Asia, Beijing 100080, China
{wuchuhan15, wufangzhao, taoqi.qt}@gmail.com
yfhuang@tsinghua.edu.cn

Abstract

Effectively finetuning pretrained language models (PLMs) is critical for their success in downstream tasks. However, PLMs may have risks in overfitting the pretraining tasks and data, which usually have gap with the target downstream tasks. Such gap may be difficult for existing PLM finetuning methods to overcome and lead to suboptimal performance. In this paper, we propose a very simple yet effective method named *NoisyTune* to help better finetune PLMs on downstream tasks by adding some noise to the parameters of PLMs before finetuning. More specifically, we propose a matrix-wise perturbing method which adds different uniform noises to different parameter matrices based on their standard deviations. In this way, the varied characteristics of different types of parameters in PLMs can be considered. Extensive experiments on both GLUE English benchmark and XTREME multilingual benchmark show *NoisyTune* can consistently empower the finetuning of different PLMs on different downstream tasks.

1 Introduction

In recent years, pretrained language models (PLMs) have achieved huge success in NLP (Qiu et al., 2020). Many PLMs such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and UniLM (Dong et al., 2019) which are pretrained from large-scale unlabeled corpus in a self-supervised way, have significantly improve various downstream tasks such as reading comprehension (Xu et al., 2019), machine translation (Brown et al., 2020), text classification (Bao et al., 2020), dialog (Wu et al., 2020) and recommendation (Wu et al., 2021) by finetuning on these tasks.

How to effectively finetune PLMs to better empower downstream tasks is an important research problem (Zheng et al., 2021). Many existing NLP methods usually directly finetune PLMs with the labeled data in downstream tasks (Sun et al., 2019). Only a few works explore more effective and robust PLM finetuning methods (Chen et al., 2020; Lee et al., 2020; Aghajanyan et al., 2021; Zhang et al., 2021; Xu et al., 2021). For example, Chen et al. (2020) proposed RecAdam that adds a penalty item to minimize the L_2 distance between the finetuned models and the pretrained models, where the penalty intensity is time-variant during finetuning. Lee et al. (2020) proposed Mixout which randomly replaces part of the parameters in the finetuned model with their original weights in the PLMs. These PLM finetuning methods mainly focus on preventing PLMs from overfitting the limited labeled data in downstream tasks. Besides the overfitting of downstream task data, a rarely studied problem is that the PLMs usually overfit the pretraining tasks and data (Qi et al., 2020), which may have significant gap with the downstream task and data. It is not easy for existing PLM finetuning methods to overcome such gap (Roberts et al., 2020), which may lead to suboptimal performance especially when labeled data in downstream tasks is insufficient.

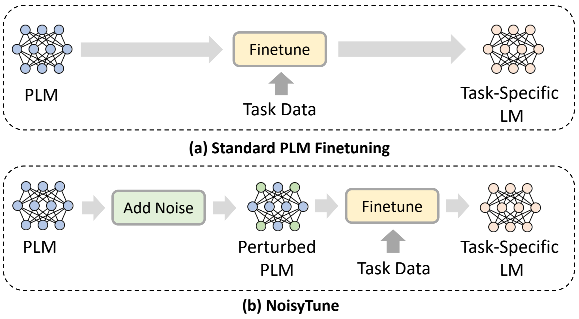


Figure 1: Schematic comparisons between standard PLM finetuning and our *NoisyTune*.

In order to handle this problem, in this paper we propose a very simple yet effective method named *NoisyTune*, which can help better finetune PLMs for downstream tasks. Different from the

[*] Corresponding author.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 680 - 685
May 22-27, 2022 ©2022 Association for Computational Linguistics

standard finetuning paradigm (Fig. 1 (a)) which directly finetunes PLMs on the downstream task data, the key idea of *NoisyTune* is to add a small amount of noise to perturb PLMs parameters before finetuning (Fig. 1 (b)). It can help prevent PLMs from overfitting the tasks and data in the pretraining stage, and reduce the gap between pretraining and downstream tasks. Since PLMs have different types of parameters which usually own different characteristics, in *NoisyTune* we use a matrix-wise perturbing method that adds uniform noise with different intensities to different parameter matrices according to their standard deviations for better adaptation. We conduct extensive experiments on two widely used NLP benchmarks, namely, GLUE (Wang et al., 2018) for English language understanding and XTREME (Hu et al., 2020) for multilingual language understanding. The results show *NoisyTune* can empower the finetuning of different PLMs on many different downstream NLP tasks to consistently achieve better performance. In addition, the results show *NoisyTune* can be easily combined with many existing PLM finetuning methods and further improve their performance.

2 NoisyTune

The goal of *NoisyTune* is for more effective finetuning of PLMs on downstream tasks. The motivation of *NoisyTune* is that PLMs are well pretrained on some unlabeled corpus with some self-supervision tasks, and they may overfit these pretraining data and tasks (Qi et al., 2020), which usually have gap with the downstream task and data. It may be difficult for PLMs to effectively adapt to downstream tasks especially when labeled data in these tasks are limited, which is usually the case. Motivated by the dueling bandits mechanism (Yue and Joachims, 2009) that adds randomness to the model for exploration, as shown in Fig. 1, we propose to add some noise to the parameters of PLMs before finetuning them on downstream tasks to do some "exploration" in parameter space and reduce the risk of overfitting the pretraining tasks and data.

PLMs usually have different kinds of parameter matrices, such as query, key, value, and feedforward network matrices (Devlin et al., 2019). Different parameter matrices in the PLMs usually have different characteristics and scales. For example, some researchers found that the self-attention parameters and the feed-forward network parameters in Transformers have very different properties, such as rank and density (Wang et al., 2020). Thus, adding unified noise to all parameter matrices in PLMs may not be optimal for keeping their good model utility. To handle this challenge, we propose a matrix-wise perturbing method that adds noise with different intensities to different parameter matrices according to their variances. Denote the parameter matrices (or scalars/vectors) in a PLM as $[\mathbf{W}_1, \mathbf{W}_2, ..., \mathbf{W}_N]$, where N is the number of parameter matrix types. Denote the perturbed version of the parameter matrix $\mathbf{W}_i$ as $\tilde{\mathbf{W}}_i$, which is computed as follows:

$$\tilde{\mathbf{W}}_i = \mathbf{W}_i + U(-\frac{\lambda}{2}, \frac{\lambda}{2}) * std(\mathbf{W}_i), \qquad (1)$$

where std stands for standard deviation. The function $U(a, b)$ represents uniform distribution noise ranged from a to b, and λ is a hyperparameter that controls the relative noise intensity.[1] We can see that in *NoisyTune* parameters in PLMs with higher variance will be added with stronger noise. In addition, in some PLMs there are some constant matrices, such as token type embeddings in RoBERTa (Liu et al., 2019). They will not be perturbed because their standard deviation is 0. It can ensure that these constant matrices will not be accidentally activated by additional noise.

NoisyTune is a simple and general plug-and-play technique that can be applied to the finetuning of any PLM on any task, simply by inserting the following PyTorch-style code before finetuning:

```
for name,para in model.named_parameters():
    model.state_dict[name][:] +=
    (torch.rand(para.size())-0.5)
    *noise_lambda*torch.std(para)
```

3 Experiments

3.1 Datasets and Experimental Settings

We conduct extensive experiments on two widely used benchmarks for PLM evaluation. The first one is GLUE (Wang et al., 2018), which is a benchmark for English language understanding that contains different tasks like natural language inference, sentiment analysis and sentence similarity evaluation. The second one is XTREME (Hu et al., 2020), which is a benchmark for multilingual language understanding. It covers 40 languages and contains

[1] Note that $U(a, b)$ is a matrix with the same shape with $\mathbf{W}_i$ rather than a scalar.

four groups of tasks, including sentence classification, structured prediction, sentence retrieval and question answering. More details of these benchmarks can refer to their original papers and official websites. Since the test labels of GLUE are not released, following (Bao et al., 2020) we report results on the dev set of GLUE. The XTREME results are evaluated on the test set. The hyperparameter λ is 0.15 on GLUE and is 0.1 on XTREME. The searching range of hyperparameters in our work are listed in Table 1.

Hyperparameters	Range
Learning rate	{7e-6, 1e-5, 2e-5, 3e-5}
Epoch	{3, 5, 7, 10, 15, 20}
Batch size	{8, 16, 32}
Noisy intensity	{0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3}

Table 1: Searching ranges of different hyperparameters in our experiments.

Following (Zheng et al., 2021), in sentence retrieval tasks we first train the models on the XNLI dataset, and then use the average of token representations produced by the hidden layer that yields the best performance. In order not to harm the alignment of token embeddings across different languages, we do not add noise to the token embeddings in multilingual PLMs. We repeat experiments 5 times with different random seeds and report the average scores.

3.2 Performance Evaluation

On the GLUE benchmark, we compare the performance of directly finetuning the base version of BERT (Devlin et al., 2019), XLNET (Yang et al., 2019), RoBERTa (Liu et al., 2019) and ELECTRA (Clark et al., 2020) with that of finetuning them after applying *NoisyTune*. On the XTREME benchmark, we compare the performance of directly finetuning both base and large versions of XLM-R (Conneau et al., 2020) with that of their variants obtained by applying *NoisyTune*. The results on these two benchmarks are shown in Tables 2 and 3, respectively. On the XTREME datasets, we report two types of results. The first one is zero-shot crosslingual transfer from English to other languages, and the second one is learning models on both English and translated data.

According to these results, *NoisyTune* can consistently improve the performance of different PLMs on different tasks in both English and multilingual settings. In addition, the performance improvement brought by *NoisyTune* is usually larger on relatively small datasets (e.g., RTE, CoLA and WNLI). These results indicate that when labeled data in downstream tasks is insufficient, it is quite difficult to effectively finetune PLMs starting from the original parameters which usually overfit the pretraining tasks and data. The experimental results validate that *NoisyTune* can properly perturb PLMs with a little noise to explore different parameter spaces and reduce the overfitting problem, making PLMs easier to be adapted to downstream tasks.

3.3 Which Noise to Use and How?

In this section we study which kind of noise is more suitable for *NoisyTune*. In addition, we explore whether our proposed matrix-wise perturbing method is better than using a unified global noise for all model parameters in PLMs. We compare five methods, including (1) *NoisyTune* without any noise; (2) *NoisyTune* with a global Gaussian noise; (3) *NoisyTune* with a global uniform noise; (4) *NoisyTune* with matrix-wise Gaussian noise; (5) *NoisyTune* with matrix-wise uniform noise. The results on GLUE are shown in Fig. 2, and the results on XTREME show similar patterns. We find that adding global noise with the same distribution to all the PLM parameters will harm the model performance. This is because different parameter matrices in PLMs have very different distributions and characteristics (Wang et al., 2020). Simply adding a unified global noise to all the parameter matrices is not optimal. The results show that matrix-wise noise is a much better choice, since the different characteristics of different parameter matrices can be taken into consideration. In addition, we find an interesting phenomenon that adding uniform noise is better than Gaussian noise. This may be because Gaussian noise has wider ranges and some extreme values may affect the model performance. Thus, we use matrix-wise uniform noise in *NoisyTune*.

3.4 Combination with Existing PLM Finetuning Methods

From Fig. 1, it is very clear that *NoisyTune* is independent of the specific PLM finetuning method, since it is applied at the stage before finetuning PLM on the task-specific data. Thus, it is very easy to combine *NoisyTune* with any kind of existing PLM finetuning method. In this section, we explore whether *NoisyTune* has the potential to empower the existing PLM finetuning techniques to achieve better performance. Here we select two

Model	**MNLI** Acc	**QNLI** Acc	**QQP** Acc	**RTE** Acc	**SST** Acc	**MRPC** Acc	**CoLA** MCC	**STS** PCC	**WNLI** Acc	Avg.
BERT	84.4	91.5	90.9	67.7	93.0	87.1	58.1	89.4	54.4	79.6
BERT+NoisyTune	84.7	91.8	91.2	68.8	93.4	88.0	59.0	90.1	56.1	80.3
XLNET	86.6	91.6	91.2	72.9	94.4	88.1	59.6	89.6	57.5	81.3
XLNET+NoisyTune	86.9	91.9	91.4	73.8	94.7	88.6	60.1	90.0	58.6	81.8
RoBERTa	87.5	92.7	91.7	77.1	94.5	90.1	62.9	90.8	59.2	82.9
RoBERTa+NoisyTune	87.8	93.1	91.9	78.8	94.9	90.6	63.6	91.1	60.3	83.6
ELECTRA	88.4	92.9	91.7	75.2	94.9	88.2	64.2	90.1	62.0	83.1
ELECTRA+NoisyTune	88.7	93.2	92.1	76.4	95.2	88.7	64.9	90.5	63.4	83.7

Table 2: Results of different methods on the GLUE dev set.

Model	**Sentence Pair**		**Structured Prediction**		**Sentence Retrieval**		**Question Answering**			
	XNLI	PAWS-X	POS	NER	BUCC	Tatoeba	XQuAD	MLQA	TyDiQA	
Metrics	Acc	Acc	F1	F1	Acc	Acc	F1/EM	F1/EM	F1/EM	Avg.
Fine-tune multilingual model on English training set (Cross-lingual Transfer)										
XLM-R_{base}	74.8	84.8	75.5	61.6	77.6	73.8	71.9/56.6	65.2/47.0	55.5/38.4	70.0
XLM-R_{base}+NoisyTune	75.2	85.1	76.0	62.1	78.2	74.5	72.3/57.1	65.5/47.4	56.0/39.2	70.5
XLM-R_{large}	79.0	86.3	72.7	62.3	79.2	76.0	76.2/60.4	71.4/53.0	65.0/45.0	72.4
XLM-R_{large}+NoisyTune	79.3	86.5	73.5	63.2	79.9	76.8	76.7/61.0	71.9/53.6	65.4/45.6	73.0
Fine-tune multilingual model on all training sets (Translate-Train-All)										
XLM-R_{base}	78.5	88.2	76.2	62.6	79.6	79.4	75.0/61.5	67.8/50.1	63.8/47.6	73.3
XLM-R_{base}+NoisyTune	78.9	88.6	76.8	63.1	80.0	79.8	75.4/61.8	68.0/50.4	64.1/48.1	73.7
XLM-R_{large}	82.3	90.3	77.3	67.3	82.5	82.7	80.0/65.6	72.9/54.4	66.3/47.6	76.4
XLM-R_{large}+NoisyTune	82.5	90.5	77.8	67.9	82.9	83.0	80.4/66.1	73.3/54.9	66.8/48.2	76.8

Table 3: Results of different methods on the XTREMRE test set.

well-known PLM finetuning for experiments, i.e., RecAdam (Chen et al., 2020) and Mixout (Lee et al., 2020). The experimental results are summarized in Fig. 3. We find that combining *NoisyTune* with existing PLM finetuning techniques can further improve their performance. This is because *NoisyTune* aims to address the overfitting of pretraining signals while these methods aim to prevent overfitting in downstream tasks. Thus, *NoisyTune* and these PLM finetuning methods are complementary, and they can be empowered by *NoisyTune* to achieve better performance.

3.5 Empirical Analysis of NoisyTune

Next, we empirically analyze why *NoisyTune* can help PLM finetuning. We compare the accuracy of BERT with and without *NoisyTune* finetuned with different percentage of samples on the MRPC dataset.[2] The results are shown in Fig. 4. We find *NoisyTune* can consistently improve PLMs under different amounts of data, especially when less training data is used. This is because the perturbed PLMs may have lower risks of overfitting the pretraining tasks and have better generalization abilities, which is especially beneficial for finetuning PLMs on downstream task with limited data.

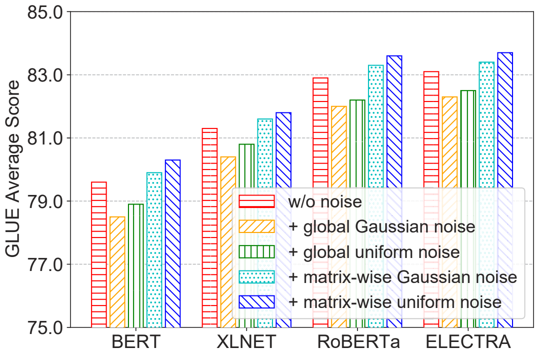


Figure 2: Different noise types and perturbing methods.

To further study the impact of *NoisyTune* on PLM finetuning, we show the relative changes of the L_1-norms of different kinds of parameters in the BERT model during finetuning on the MRPC dataset in Fig. 5.[3] Since the noise we added to PLMs in *NoisyTune* is zero-mean uniform noise, the absolute parameter L_1-norm will not change too much. However, we can see that the relative change of L_1-norms becomes smaller when *NoisyTune* is applied, which indicates that the PLMs can find the (sub)optimal parameters for downstream

[2] We observe similar patterns on other datasets.

[3] The patterns on other datasets are similar.

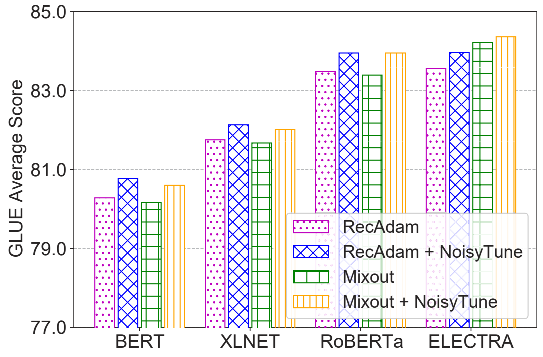


Figure 3: *NoisyTune* can empower many existing PLM finetuning methods to achieve better performance.

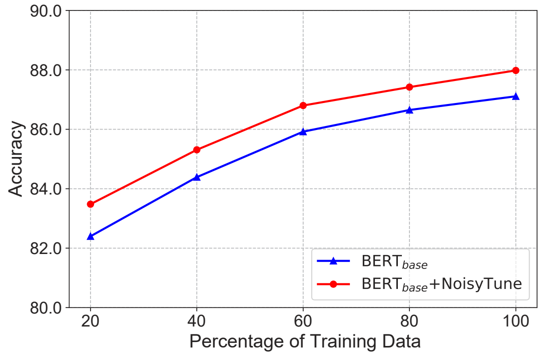


Figure 4: Influence of *NoisyTune* on finetuning.

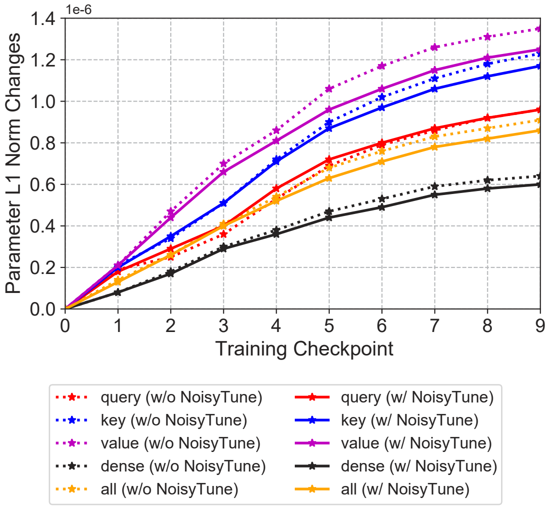


Figure 5: Relative changes of the L_1-norm of different types of parameters in PLM during finetuning.

tasks more easily. This result validates directly finetuning PLMs may need more updates to adapt to downstream tasks, which is due to the overfitting of pretraining tasks, and *NoisyTune* can provide a simple way to alleviate this problem and help finetune PLMs on downstream tasks more effectively.

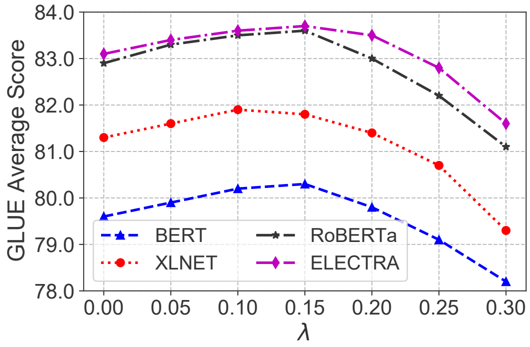


Figure 6: Influence of noise intensity λ.

3.6 Hyperparameter Analysis

We study the influence of the most important hyperparameter in *NoisyTune*, i.e., λ, which controls the relative noise intensity. The average GLUE scores w.r.t. different λ values are shown in Fig. 6. We find that when λ is too small or too large, the performance is not optimal. This is because when λ is too small, it is difficult for PLMs to do parameter space exploration and overcome the overfitting problem. While when λ is too large, the useful pretrained knowledge in PLMs may be overwhelmed by random noise. Values between 0.1 and 0.15 are more suitable for *NoisyTune* on the GLUE datasets.

4 Conclusion

In this paper, we propose a very simple but effective method named *NoisyTune*, which can help better finetune PLMs on downstream tasks by adding a little noise to them before finetuning. In *NoisyTune*, we propose a matrix-wise perturbing method that adds noise with different intensities to different kinds of parameter matrices in PLMs according to their variances. *NoisyTune* is a very general method, and is PLM model agnostic, downstream task agnostic, and finetuning method agnostic. Extensive experiments on both monolingual GLUE benchmark and multilingual XTREME benchmark demonstrate *NoisyTune* can consistently empower the finetuning of different PLMs on various downstream tasks to achieve better performance.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant numbers U1936216, U1936208, and 61862002, and the research initiation project of Zhejiang Lab (No. 2020LC0PI01).

References

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. 2021. Better fine-tuning by reducing representational collapse. In *ICLR*.

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, et al. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *ICML*, pages 642–652. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *NeurIPS*, 33:1877–1901.

Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. 2020. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. In *EMNLP*, pages 7870–7881.

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

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *ACL*, pages 8440–8451.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *NIPS*, pages 13063–13075.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *ICML*, pages 4411–4421. PMLR.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2020. Mixout: Effective regularization to finetune large-scale pretrained language models. In *ICLR*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequencepre-training. In *EMNLP Findings*, pages 2401–2410.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *EMNLP*, pages 5418–5426.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *CCL*, pages 194–206. Springer.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *BlackboxNLP*, pages 353–355.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Chien-Sheng Wu, Steven CH Hoi, Richard Socher, and Caiming Xiong. 2020. Tod-bert: Pre-trained natural language understanding for task-oriented dialogue. In *EMNLP*, pages 917–929.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Empowering news recommendation with pre-trained language models. In *SIGIR*, pages 1652–1656. ACM.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2019. Bert post-training for review reading comprehension and aspect-based sentiment analysis. In *NAACL-HLT*, pages 2324–2335.

Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. 2021. Raise a child in large language model: Towards effective and generalizable fine-tuning. In *EMNLP*, pages 9514–9528.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, pages 5753–5763.

Yisong Yue and Thorsten Joachims. 2009. Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML*, pages 1201–1208.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample bert fine-tuning. In *ICLR*.

Bo Zheng, Li Dong, Shaohan Huang, Wenhui Wang, Zewen Chi, Saksham Singhal, Wanxiang Che, Ting Liu, Xia Song, and Furu Wei. 2021. Consistency regularization for cross-lingual fine-tuning. In *ACL-IJCNLP*, pages 3403–3417.

Adjusting the Precision-Recall Trade-Off with Align-and-Predict Decoding for Grammatical Error Correction

Xin Sun Houfeng Wang
MOE Key Lab of Computational Linguistics, School of Computer Science, Peking University
{sunx5,wanghf}@pku.edu.cn

Abstract

Modern writing assistance applications are always equipped with a Grammatical Error Correction (GEC) model to correct errors in user-entered sentences. Different scenarios have varying requirements for correction behavior, e.g., performing more precise corrections (high precision) or providing more candidates for users (high recall). However, previous works adjust such trade-off only for sequence labeling approaches. In this paper, we propose a simple yet effective counterpart – Align-and-Predict Decoding (APD) for the most popular sequence-to-sequence models to offer more flexibility for the precision-recall trade-off. During inference, APD aligns the already generated sequence with input and adjusts scores of the following tokens. Experiments in both English and Chinese GEC benchmarks show that our approach not only adapts a single model to precision-oriented and recall-oriented inference, but also maximizes its potential to achieve state-of-the-art results. Our code is available at https://github.com/AutoTemp/Align-and-Predict.

1 Introduction

Modern writing assistance applications (e.g., Microsoft Office Word[1], Google Docs[2] and Grammarly[3]) always contain Grammatical Error Correction (GEC) modules (Ge et al., 2018; Omelianchuk et al., 2020; Stahlberg and Kumar, 2021) to correct errors in user-entered sentences. Such applications usually require GEC models to perform different correction tendencies and behaviors according to practical scenarios and user preferences (Chen et al., 2020). For instance, as shown in Table 1, conservative GEC models provide precise corrections with high confidence and avoid unnecessary edits for better user experience. In contrast, aggressive GEC models could provide more correction candidates to users or a following decision system for further measurement.

Input	I believe we have the experience of suddenly forget how to write a word we should know.
Conservative GEC	I believe we have the experience of suddenly [forgetting]$_0$ how to write a word we should know.
Aggressive GEC	I believe [most of us]$_0$ [had]$_1$ the [experiences]$_2$ of suddenly [forgetting]$_3$ how to write a word [that]$_4$ we should know.

Table 1: Examples of corrections generated by the conservative (precision-oriented) and aggressive (recall-oriented) GEC models. The rewritten tokens are within the blue blocks. Conservative GEC tends to adhere to the input sentence, while aggressive GEC provides more edited spans.

Although recent studies witness the tremendous success of sequence-to-sequence (seq2seq) generation approaches in GEC, the trade-off of these two tendencies still largely depends on the pre-defined model architecture, training data and labor-consuming post-processing (Liang et al., 2020). Hotate et al. (2020) proposes a diverse local beam search method to obtain diverse corrections but is specifically designed for copy-augmented GEC models and cannot perform precision-oriented decoding. Instead of seq2seq generation, Omelianchuk et al. (2020) proposes an efficient sequence tagger for GEC by token-level transformations to map input tokens to target corrections. They introduce two confidence thresholds for inference to force the model to perform more precise corrections. Chen et al. (2020) first identifies incorrect spans with a tagging model and then sets a probability threshold to adjust the precision-recall trade-off.

Inspired by these lightweight tweaking methods for sequence labeling approaches, we propose a simple yet effective counterpart – Align-

[1] https://www.microsoft.com/en-us/microsoft-365/word

[2] https://www.google.com/docs/about

[3] https://www.grammarly.com

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 686 - 693
May 22-27, 2022 ©2022 Association for Computational Linguistics

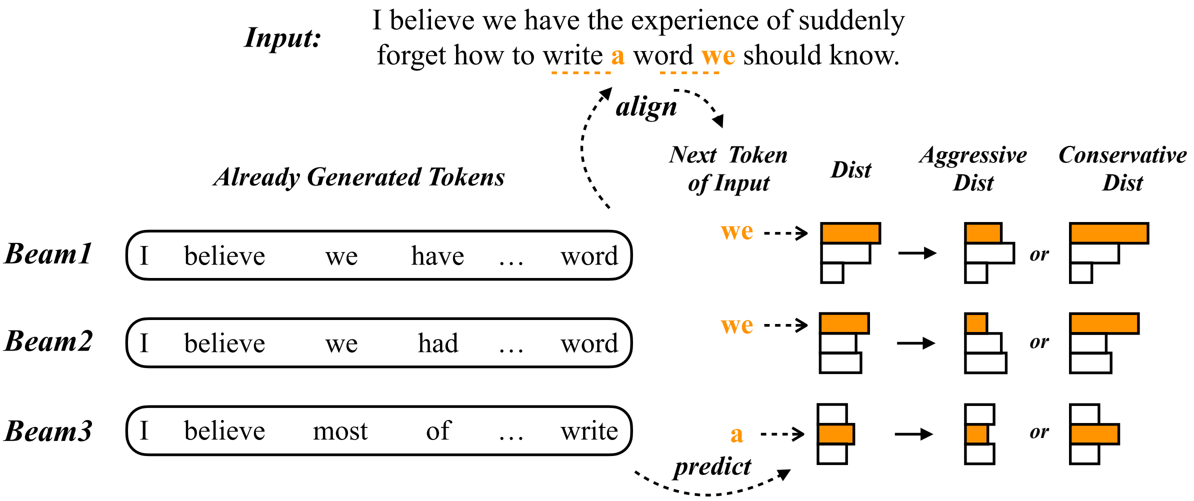


Figure 1: The overview of align-and-predict decoding. Our approach aligns already generated sequences with input tokens for all hypotheses and re-scores the next tokens (i.e., *we* and *a* highlighted in orange) at the aligned positions (highlighted with the orange dashed lines). Specifically, since the suffixes of hypothesis are *word*, *word* and *write*, which are unique in the input sentence, we select the corresponding following words – *we*, *we* and *a*. By decreasing or increasing corresponding scores (rectangles highlighted in orange), our approach adapts the precision-recall trade-off to aggressive or conservative inference. **Dist** denotes **Distribution**.

and-Predict Decoding (APD) for the seq2seq GEC models. Our approach could not only adapt the precision-recall trade-off of a single seq2seq GEC model to various application scenarios, but also be used as a simple trick to improve its overall $F_{0.5}$ performance.

During inference, APD aligns the already generated sequence with the input tokens to specify the position which the model has reached. By tweaking the score of the next token, the model changes its preference between copy and edit operation, leading to a different degree of adherence to the input sentence. The experimental results in both English and Chinese GEC benchmarks show our approach could effectively control the precision-recall trade-off and achieve state-of-the-art results. Our contributions are summarized as follows:

- We propose a novel and simple decoding approach, allowing us to adapt the precision-recall trade-off of a seq2seq GEC model.
- Our methods achieve state-of-the-art results in both English and Chinese GEC benchmarks.

2 Align-and-Predict Decoding

Beam search (Lowerre, 1976; Och and Ney, 2004; Sutskever et al., 2014) is a widely used algorithm for decoding sequences on all generation tasks, such as translation (Vaswani et al., 2017; Ott et al., 2018), dialogue (Kulikov et al., 2019), etc. Multiple modifications to beam search that force the outputs to include pre-defined lexical constraints (i.e., words and phrases) have been proposed (Hokamp and Liu, 2017; Hu et al., 2019).

Fortunately, the input and output sentences of GEC overlap significantly and the input tokens are natural constraints for correction generation. This assumption is an objective characteristic of GEC and has been made in many previous works (Zhao et al., 2019; Malmi et al., 2019; Stahlberg and Kumar, 2020; Sun et al., 2021). Thus, we propose a novel decoding approach – Align-and-Predict Decoding (APD), which leverage the characteristic of GEC to adjust behavior and tendencies of inference. The overview of APD is shown in Figure 1.

Given an input sentence $\boldsymbol{x} = (x_1, \dots, x_n)$, we maintain K hypotheses at the time step t during inference as beam search does:

$$\begin{aligned} \boldsymbol{H}_t &= \{\boldsymbol{h}^1_{\le t}, ..., \boldsymbol{h}^K_{\le t}\} \\ &= \{(y^1_1, ..., y^1_t), ..., (y^K_1, ..., y^K_t)\} \end{aligned} \quad (1)$$

where $\boldsymbol{h}^i_{\le t}, i \in [1, K]$ denotes the i-th hypothesis with t already generated tokens.

Since the output of GEC is highly constrained by the input sequence, we assume that $\boldsymbol{h}^i_{\le t}$ should be almost the same as part of the input sentence $\boldsymbol{x}$. Then, we match the suffix of each hypothesis $\boldsymbol{h}^i_{\le t}$ with the input $\boldsymbol{x}$ to identify the position which the inference has reached. If there exists a unique substring $x_{k-j}, ..., x_k (j \geq 0)$ of the input $\boldsymbol{x}$ identical to the suffix $y^i_{t-j}, ..., y^i_t$, the next token of the hypothesis $\boldsymbol{h}^i_{\leq t}$ is very likely to be x_{k+1}, which we store in the set N^i_t. [4] Formally,

$$N^i_t = \begin{cases} \{x_{k+1}\} & \exists! k, x_{k-j...k} = y^i_{t-j...t}; \\ \emptyset & otherwise. \end{cases} \tag{2}$$

As beam search does, we expand current hypotheses and construct possible candidates for the next time step $t+1$ with all tokens in the vocabulary. The candidate $\hat{\boldsymbol{h}}^i_{t,v}$ of the i-th hypothesis is obtained as follows:

$$\hat{\boldsymbol{h}}^i_{t,v} = \textsc{Cat}(\boldsymbol{h}^i_{\leq t}, v) = (y^i_1, ..., y^i_t, v) \tag{3}$$

where we concatenate the already generated sequence $\boldsymbol{h}^i_{\leq t}$ with any token v in the vocabulary. The corresponding score is calculated by:

$$\begin{aligned} \textsc{Score}(\hat{\boldsymbol{h}}^i_{t,v}) &= \textsc{Score}(\boldsymbol{h}^i_{\leq t}) \\ &+ w^i_{t,v} \cdot \log P(v | y^i_1, ..., y^i_t) \end{aligned} \tag{4}$$

where P is the output distribution predicted by the seq2seq GEC model and $w^i_{t,v}$ is a penalty factor that depends on whether the token v is the next token x_{k+1} at the aligned position. Specifically,

$$w^i_{t,v} = \begin{cases} \lambda & v \in N^i_t \\ 1.0 & v \notin N^i_t \end{cases} \tag{5}$$

where λ is a hyperparameter to control the adherence to the input sequence. If $\lambda > 1.0$, inference penalizes the score of the original next token and tends to perform modification; [5] if $\lambda < 1.0$, it is likely to copy the token. The new hypotheses are selected by:

$$\boldsymbol{H}_{t+1} = \underset{i,v}{\arg\text{topK}}(\textsc{Score}(\hat{\boldsymbol{h}}^i_{t,v})) \tag{6}$$

3 Experiments

3.1 Experimental Setting

We conduct our experiments in the restricted training setting of BEA-2019 GEC shared task (Bryant et al., 2019), with Lang-8 Corpus of Learner English (Mizumoto et al., 2011), NUCLE (Dahlmeier et al., 2013), FCE (Yannakoudakis et al., 2011) and W&I+LOCNESS (Granger; Bryant et al., 2019) as training data. We use BEA-2019 development set to choose the best model and select λ between 0.1 and 2.5 with 0.05 intervals based on $F_{0.3}$, $F_{0.5}$ and $F_{1.0}$ for precision-oriented, balance and recall-oriented models, respectively[6]. We evaluate the performance on BEA-2019 test set by ERRANT (Bryant et al., 2017).

Model	BEA-2019		
	P	R	$F_{0.5}$
Omelianchuk et al. (2020)	79.2	53.9	72.4
Kaneko et al. (2020)	67.1	60.1	65.6
Wan et al. (2020)	66.9	60.6	65.5
Lichtarge et al. (2020)	67.6	62.5	66.5
Stahlberg and Kumar (2021)	72.1	64.4	70.4
gT5 xxl (Rothe et al., 2021)	-	-	69.8
T5 xl (Rothe et al., 2021)♣	-	-	73.9
T5 xxl (Rothe et al., 2021)♣	-	-	75.9
Yuan et al. (2021)	73.3	61.5	70.6
Sun et al. (2021)	-	-	72.9
Seq2Seq (w/o pretraining)	57.4	41.8	53.4
+ Precision-oriented($\lambda = 0.45$)	**63.6**	32.9	53.6
+ Recall-oriented($\lambda = 1.95$)	51.4	**47.6**	50.5
+ Balance($\lambda = 0.75$)	59.8	39.0	**54.0**
Seq2Seq (w/ pretraining)	66.7	62.3	65.8
+ Precision-oriented($\lambda = 0.20$)	**78.5**	43.0	67.4
+ Recall-oriented($\lambda = 1.85$)	61.9	**65.6**	62.6
+ Balance($\lambda = 0.45$)	72.6	55.4	**68.3**
12+2 BART (Sun et al., 2021)	76.1	65.6	73.8
+ Precision-oriented($\lambda = 0.25$)	**88.1**	44.8	73.8
+ Recall-oriented($\lambda = 2.50$)	67.7	**72.0**	68.5
+ Balance($\lambda = 0.75$)	78.7	63.2	**75.0**

Table 2: Performance of our approach compared with previous work in BEA-2019 test set. Note that we only compare single models **without ensemble**. λ is selected based on BEA-2019 development set. It is notable that the models with ♣ are not comparable here because they use a much larger model capacity (up to 11B parameters), and their training data is different from ours: they use cleaned LANG-8 Corpus.

To validate the effectiveness of our approach for the state-of-the-art seq2seq GEC models, we follow previous work (Grundkiewicz et al., 2019; Zhang et al., 2019) to construct 300M error-corrected sentence pairs in the same way for pretraining. We use Transformer (big) model (Vaswani et al., 2017) in the fairseq[7] and a vocabulary with size of 32K Byte Pair Encoding (Sennrich et al., 2016) tokens. We also use one of the models trained by the prior work (Sun et al., 2021) which utilizes a pretrained model BART (Lewis et al., 2019) to initialize a GEC model which has a 12-layer encoder and 2-

[4] We use a lookup table (i.e., dictionary) to record the next token of n-grams (e.g., $n = 1$) in the source sentence.

[5] It is notable that λ tweaks $\log P(v)$ which is negative rather than $P(v)$. When $\lambda > 1.0$, $\lambda \cdot \log P(v)$ becomes smaller which penalizes the score of v.

[6] $F_\beta = (1+\beta^2) \cdot \frac{\text{precision}\cdot\text{recall}}{(\beta^2\cdot\text{precision})+\text{recall}}$, where recall is considered β times as important as precision. Compared with $F_{0.5}$ which is the official metric for GEC, $F_{0.3}$ and $F_{1.0}$ pay more attention to precision and recall, respectively.

[7] https://github.com/pytorch/fairseq

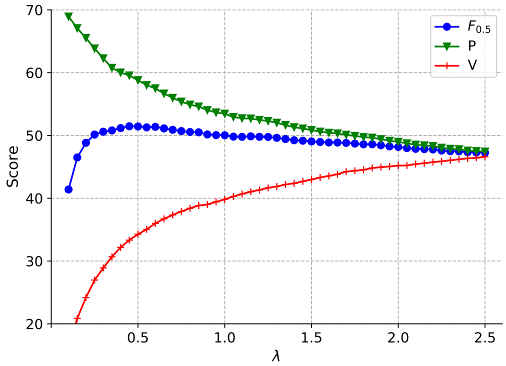


Figure 2: The performance of the *seq2seq model (w/ pretraining)* over varying λ in BEA-2019 dev set.

Model	NLPCC-2018		
	P	R	$F_{0.5}$
Fu et al. (2018)	35.2	18.6	29.9
Zhou et al. (2018)	41.0	13.8	29.4
Ren et al. (2018)	47.2	12.6	30.6
Wang et al. (2020b)	41.9	22.0	35.5
Wang et al. (2020a)	39.4	22.8	34.4
Zhao and Wang (2020)	44.4	22.4	37.0
Our Implementation	41.5	25.7	36.9
+ Precision-oriented($\lambda = 0.25$)	**52.9**	12.8	32.6
+ Recall-oriented($\lambda = 2.50$)	34.2	**34.6**	34.3
+ Balance($\lambda = 0.75$)	44.6	22.7	**37.4**

Table 3: Performance of our approach in the NLPCC-2018 Chinese benchmark. Note that the models compared here are not pretrained, except for Wang et al. (2020a).

layer decoder, following Li et al. (2021).

In addition, we evaluate our approach on NLPCC-18 Chinese GEC shared task (Zhao et al., 2018) by official Max-Match scorer[8] to prove our approach is language-independent. We use a base Transformer model and construct a character-level vocabulary consisting of 7K tokens. We train the model using MaskGEC (Zhao and Wang, 2020).

The models decode with a beam size of 5. We show more details of training in the Appendix.

3.2 Experimental Result

As shown in Table 2, our approach can control the precision-recall trade-off of inference for any seq2seq GEC models by tweaking a single hyperparameter λ. After inference tweaks, pretrained GEC models could achieve much better precision with comparable or even better overall performance. For instance, our approach increases the precision of pretrained models by over 10 points. In contrast, the recall improvement is smaller than precision,

Input	In my opinion, the car isn't necessary when you have crashed in the street, in that moment you realized the importance of a public transport.
$\lambda = 0.20$	In my opinion, the car isn't necessary when you have crashed in the street[.]$_0$ [At]$_1$ that moment you realized the importance of []$_2$ public transport.
$\lambda = 1.85$	In my opinion, [a]$_0$ car isn't necessary when you have crashed in the street[.]$_1$ [At]$_2$ that moment [,]$_3$ you [realize]$_4$ the importance of []$_5$ public transport .
Input	we can see that there are lots of serious and frequently weather disaster happened in decades, such as typhoon, hurricane, wild fire and mud slide.
$\lambda = 0.20$	we can see that there are lots of serious and frequently weather disaster happened in decades, such as typhoon, hurricane, wild fire and mud slide.
$\lambda = 0.35$	we can see that there are lots of serious and frequently weather [disasters]$_0$ [that]$_1$ [have]$_2$ happened in decades, such as typhoon, hurricane, wild fire and mud slide.
$\lambda = 1.85$	[We]$_0$ can see that [many]$_1$ serious and [frequent]$_2$ weather [disasters]$_3$ [have]$_4$ happened in decades, such as [typhoons]$_5$, [hurricanes]$_6$, [wildfires]$_7$ and [mudslides]$_8$.

Table 4: Examples of corrections generated by *seq2seq model (w/ pretraining)* with different λ. The rewritten tokens are within the blue blocks.

i.e., an increment of about 6 points for pretrained models, since it depends mainly on error-corrected patterns that the model itself has learned. The final system has achieved competitive performance (73.8 $F_{0.5}$) and align-and-predict decoding improves it to a new state-of-the-art result – 75.0 $F_{0.5}$ in the BEA-2019 test set by a slight tendency towards precision.

We further look into the performance of the pretrained seq2seq model over varying λ in BEA-2019 development set, which is shown in Figure 2. It is obvious that the conservative inference ($\lambda < 1.0$) with fewer edits tends to achieve higher precision since it only provides the most confident corrections, while recall of aggressive inference ($\lambda > 1.0$) has an upper bound. This is because the motivation of our approach is to simply display error-corrected patterns that the model has learned with different orientation rather than to improve its capability and complement more patterns. Meanwhile, it is observed that $F_{0.5}$ does not peak around $\lambda = 1.0$, which makes it possible to adapt the precision-recall trade-off for better overall performance.

As shown in Table 3, our approach also performs well in Chinese GEC, which demonstrates that it is language-independent. We present concrete exam-

[8] https://github.com/nusnlp/m2scorer

ples with different λ in our validation set in Table 4. It is consistent with our intuition that with larger λ, the inference tends to heavily edit the input tokens; on the other hand, it adheres to the input sequence with smaller λ.

4 Conclusion

We propose a novel language-independent decoding approach to offer more flexibility to adjust the precision-recall trade-off of inference for seq2seq GEC models, making it adaptive to various real-world application scenarios. It can not only adapt a single model to precision-oriented and recall-oriented inference, but also be used as a simple trick for better overall performance. On both English and Chinese GEC benchmarks, our approach further improves the state-of-the-art seq2seq GEC model by precision-recall trade-off. In the future, we plan to apply it to other sentence rewriting tasks, such as paraphrasing and style transfer.

Acknowledgments

We thank all the reviewers for their valuable comments to improve our paper. We thank Tao Ge in Microsoft Research Asia for the valuable suggestions. The work is supported by National Natural Science Foundation of China under Grant No.62036001 and PKU-Baidu Fund (No.2020BD021). The corresponding author of this paper is Houfeng Wang.

References

Christopher Bryant, Mariano Felice, Øistein E Andersen, and Ted Briscoe. 2019. The bea-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 793–805.

Mengyun Chen, Tao Ge, Xingxing Zhang, Furu Wei, and Ming Zhou. 2020. Improving the efficiency of grammatical error correction with erroneous span detection and correction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7162–7169.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner english: The nus corpus of learner english. In *Proceedings of the eighth workshop on innovative use of NLP for building educational applications*, pages 22–31.

Kai Fu, Jin Huang, and Yitao Duan. 2018. Youdao's winning solution to the nlpcc-2018 task 2 challenge: a neural machine translation approach to chinese grammatical error correction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 341–350. Springer.

Tao Ge, Furu Wei, and Ming Zhou. 2018. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1055–1065.

Sylviane Granger. *The computer learner corpus: a versatile new source of data for SLA research.*

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 252–263.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546.

Kengo Hotate, Masahiro Kaneko, and Mamoru Komachi. 2020. Generating diverse corrections with local beam search for grammatical error correction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2132–2137.

J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4248–4254.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Ilia Kulikov, Alexander Miller, Kyunghyun Cho, and Jason Weston. 2019. Importance of search and evaluation strategies in neural dialogue modeling. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 76–87.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yanyang Li, Ye Lin, Tong Xiao, and Jingbo Zhu. 2021. An efficient transformer decoder with compressed sub-layers. *arXiv preprint arXiv:2101.00542*.

Deng Liang, Chen Zheng, Lei Guo, Xin Cui, Xiuzhang Xiong, Hengqiao Rong, and Jinpeng Dong. 2020. Bert enhanced neural machine translation and sequence tagging model for chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 57–66.

Jared Lichtarge, Chris Alberti, and Shankar Kumar. 2020. Data weighted training strategies for grammatical error correction. *Transactions of the Association for Computational Linguistics*, 8:634–646.

Bruce T Lowerre. 1976. *The harpy speech recognition system.* Carnegie Mellon University.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5054–5065.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning sns for automated japanese error correction of second language learners. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 147–155.

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational linguistics*, 30(4):417–449.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector–grammatical error correction: Tag, not rewrite. *arXiv preprint arXiv:2005.12592*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. *arXiv preprint arXiv:1806.00187*.

Hongkai Ren, Liner Yang, and Endong Xun. 2018. A sequence to sequence learning for chinese grammatical error correction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 401–410. Springer.

Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. *arXiv preprint arXiv:2106.03830*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Felix Stahlberg and Shankar Kumar. 2020. Seq2edits: Sequence transduction using span-level edit operations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5147–5159.

Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 37–47.

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. *arXiv preprint arXiv:2106.04970*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

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

Zhaohong Wan, Xiaojun Wan, and Wenguang Wang. 2020. Improving grammatical error correction with data augmentation by editing latent representation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2202–2212.

Chencheng Wang, Liner Yang, Yingying Wang, Yongping Du, and Erhong Yang. 2020a. Chinese grammatical error correction method based on transformer enhanced architecture. *Journal of Chinese Information Processing*, 34(6):106.

Hongfei Wang, Michiki Kurosawa, Satoru Katsumata, and Mamoru Komachi. 2020b. Chinese grammatical correction using bert-based pre-trained model. *arXiv preprint arXiv:2011.02093*.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 180–189.

Zheng Yuan, Shiva Taslimipoor, Christopher Davis, and Christopher Bryant. 2021. Multi-class grammatical error detection for correction: A tale of two systems.

In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8722–8736.

Yi Zhang, Tao Ge, Furu Wei, Ming Zhou, and Xu Sun. 2019. Sequence-to-sequence pre-training with data augmentation for sentence rewriting. *arXiv preprint arXiv:1909.06002*.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 156–165.

Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. Overview of the nlpcc 2018 shared task: Grammatical error correction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 439–445. Springer.

Zewei Zhao and Houfeng Wang. 2020. Maskgec: Improving neural grammatical error correction via dynamic masking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1226–1233.

Junpei Zhou, Chen Li, Hengyou Liu, Zuyi Bao, Guangwei Xu, and Linlin Li. 2018. Chinese grammatical error correction using statistical and neural models. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 117–128. Springer.

A Hyper-parameters

The hyper-parameters for Chinese GEC are listed in Table 5. The hyper-parameters of training the models for English GEC are listed in Table 6 and Table 7.

Configurations	Values
Train From Scratch	
Model Architecture	Transformer (base)
Training Strategy	MaskGEC (Zhao and Wang, 2020)
Devices	4 Nvidia V100 GPU
Max tokens per GPU	5120
Update Frequency	[2, 4]
Optimizer	Adam (β_1=0.9, β_2=0.98, ϵ=1 $\times 10^{-8}$) (Kingma and Ba, 2014)
Learning rate	[$5 \times 10^{-4}, 7 \times 10^{-4}$]
Learning rate scheduler	inverse sqrt
Warmup	4000
weight decay	0.0
Loss Function	label smoothed cross entropy (label-smoothing=0.1) (Szegedy et al., 2016)
Dropout	0.3

Table 5: Hyper-parameters values for Chinese GEC.

Configurations	Values
Pretrain	
Model Architecture	Transformer (big)
Number of epochs	10
Devices	8 Nvidia V100 GPU
Max tokens per GPU	5120
Update Frequency	8
Learning rate	3×10^{-4}
Optimizer	Adam (β_1=0.9, β_2=0.98, ϵ=1 $\times 10^{-8}$)
Learning rate scheduler	inverse sqrt
Weight decay	0.0
Loss Function	label smoothed cross entropy (label-smoothing=0.1)
Warmup	8000
Dropout	0.3
Fine-tune	
Number of epochs	60
Devices	4 Nvidia V100 GPU
Update Frequency	4
Learning rate	3×10^{-4}
Warmup	4000
Dropout	0.3

Table 6: Hyper-parameters values of pretraining and fine-tuning for English GEC.

Configurations	Values
Pretrain	
Model Architecture	BART 12+2 Init
Number of steps	400000 with early stopping
Devices	32 Nvidia V100 GPU
Max tokens per GPU	8000
Update Frequency	4
Learning rate	1×10^{-4}
Optimizer	Adam (β_1=0.9, β_2=0.999, ϵ=1 $\times 10^{-8}$)
Learning rate scheduler	polynomial decay
Weight decay	0.01
Loss Function	label smoothed cross entropy (label-smoothing=0.1)
Warmup	16000
Dropout	0.3
Fine-tune	
Training Strategy	Multi-stage fine-tuning (Stahlberg and Kumar, 2020)
Devices	8 Nvidia V100 GPU
Learning rate	5×10^{-5}
Warmup	4000
Dropout	0.2

Table 7: Hyper-parameters values of the BART-initialized model for English GEC.

Model	**Time (in second)**		
	1	16	64
Seq2Seq (w/ pretraining)	218	37	20
+ $\lambda = 0.20$	225	41	23
+ $\lambda = 1.85$	229	42	23

Table 8: The total inference time of the *seq2seq model (w/ pretraining)* under various batch sizes (1/16/64) using 1 NVIDIA TITAN RTX GPU with CUDA 11.1 in the first 1000 sentences of the BEA-2019 dev set.

B Efficiency

Table 8 shows the total latency of the *seq2seq model (w/ pretraining)* under various batch sizes. Our approach incurs about 5% extra latency in the online inference setting (i.e., batch size=1) and is suitable for practical GEC systems.

On the Effect of Isotropy on VAE Representations of Text

Lan Zhang[♠] **Wray Buntine**[♠♡] **Ehsan Shareghi**[♠♣]
♠ Department of Data Science & AI, Monash University
♡ College of Eng. and Comp. Sc., VinUniversity
♣ Language Technology Lab, University of Cambridge
{lan.zhang, wray.buntine, ehsan.shareghi}@monash.edu

Abstract

Injecting desired geometric properties into text representations has attracted a lot of attention. A property that has been argued for, due to its better utilisation of representation space, is isotropy. In parallel, VAEs have been successful in areas of NLP, but are known for their sub-optimal utilisation of the representation space. To address an aspect of this, we investigate the impact of injecting isotropy during training of VAEs. We achieve this by using an isotropic Gaussian posterior (IGP) instead of the ellipsoidal Gaussian posterior. We illustrate that IGP effectively encourages isotropy in the representations, inducing a more discriminative latent space. Compared to vanilla VAE, this translates into a much better classification performance, robustness to input perturbation, and generative behavior. Additionally, we offer insights about the representational properties encouraged by IGP.[1]

1 Introduction

In recent years, with the success facilitated by pretrained representations across various NLP tasks, more attention has been placed on studying and utilising the geometric properties of learned representations. A phenomena that has been studied more recently in this direction is anisotropy (Ethayarajh, 2019), indicating a sub-optimal property where the learned embeddings only utilise a small subset of the representation space. Various methods have been proposed to rectify this and encourage the representations to be more discriminative or to exploit the representation dimensions more effectively (Liu et al., 2021; Gao et al., 2021; Li et al., 2020a; Su et al., 2021; Mu and Viswanath, 2018).

In parallel, Variational Autoencoders (VAEs) (Kingma and Welling, 2014) have been widely used in various areas of NLP, from representation learning for downstream tasks (Li et al., 2020b; Wei and Deng, 2017), to generation (Prokhorov et al., 2019; Bowman et al., 2016), representational sparsity and disentanglement (Prokhorov et al., 2021; Zhang et al., 2021), and semi-supervised learning (Zhu et al., 2021; Choi et al., 2019; Yin et al., 2018; Xu et al., 2017). In recent years, most of the developments around VAEs have focused on avoiding the posterior collapse (Bowman et al., 2016) which leads to learning sub-optimal representations (Havrylov and Titov, 2020; Fu et al., 2019; Li et al., 2019; Dieng et al., 2019; He et al., 2019; Higgins et al., 2017; Yang et al., 2017; Bowman et al., 2016). Despite the success of these techniques, a non-collapsed VAE still utilises the representation space sub-optimally (Prokhorov et al., 2019; He et al., 2019; Burda et al., 2016), as very commonly the learned representations do not fully utilise the latent space to encode information.

In this paper we bridge between the two lines of research by injection isotropy in the latent space of VAEs. Such property could be encouraged by using an Isotropic Gaussian Posterior (IGP) which involves a simple modification of VAEs. An Isotropic Gaussian distribution, $\mathcal{N}(\boldsymbol{\mu}, \sigma^2\boldsymbol{I})$, is similar to vanilla VAE's posterior with the exception that all dimensions share the same unified variance. Tying the variances would encourage encoder of VAEs towards the extreme where *all* dimensions are either active or inactive.[2]

Our experimental findings indicate that, compared to vanilla VAE, the use of IGP is effective in both increasing dimension activation and injecting isotropy in the learned representation space. We observe that isotropy results in a more discriminative representation space which is much more suited for classification tasks and robust to input perturbation.

[1] Code and datasets are available at https://github.com/lanzhang128/IGPVAE

[2] A dimension u is defined to be active if $A_u = \mathrm{Cov}_{\mathbf{x}}(\mathbb{E}_{u\sim q(u|\mathbf{x})}[u])$ is larger than 0.01, where Cov denotes covariance (Burda et al., 2016).

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 694 - 701
May 22-27, 2022 ©2022 Association for Computational Linguistics

Our generative experiment for sentence completion suggests that the VAE trained with IGP is more capable of maintaining semantic cohesiveness.

2 Isotropic Gaussian Posterior (IGP)

Variational Autoencoder (VAE). Let $\mathbf{x}$ denote datapoints in data space and $\mathbf{z}$ denote latent variables in the latent space, and assume the datapoints are generated by the combination of two random processes: The first random process is to sample a point $\mathbf{z}^{(i)}$ from the latent space in VAEs with prior distribution of $\mathbf{z}$, denoted by $p(\mathbf{z})$. The second random process is to generate a point $\mathbf{x}^{(i)}$ from the data space, denoted by $p(\mathbf{x}|\mathbf{z}^{(i)})$. VAE uses a combination of a probabilistic encoder $q_\phi(\mathbf{z}|\mathbf{x})$ and decoder $p_\theta(\mathbf{x}|\mathbf{z})$, parameterised by ϕ and θ, to learn this statistical relationship between $\mathbf{x}$ and $\mathbf{z}$. VAE is trained by maximizing the lower bound of the logarithmic data distribution $\log p(\mathbf{x})$, called evidence lower bound (ELBO), $\mathcal{L}(\phi, \theta; \mathbf{x})$:

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] - \mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

The first term of objective function is the expectation of the logarithm of data likelihood under the posterior distribution of z. The second term is KL-divergence, measuring the distance between the recognition distribution $q_\phi(\mathbf{z}|\mathbf{x})$ and the prior distribution $p(\mathbf{z})$ and can be seen as a regularisation.

In the presence of auto-regressive and powerful decoders, a common optimisation challenge of training VAEs in text modelling is called posterior collapse, where the learned posterior distribution $q_\phi(\mathbf{z}|\mathbf{x})$, collapses to the prior $p(\mathbf{z})$. Several strategies have been proposed to alleviate this problem (Bowman et al., 2016; Havrylov and Titov, 2020; Fu et al., 2019; He et al., 2019). In this work, we follow Prokhorov et al. (2019), $\mathcal{L}(\phi, \theta; \mathbf{x})$:

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] - \beta|\mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) - C|$$

where C is a positive real value which represents the target KL-divergence term value and β indicates the regularisation strength. We set $\beta = 1$ to make sure the weights of the two terms balance, noting that it acts as a Lagrange Multiplier (Boyd and Vandenberghe, 2004). This also has an information-theoretic interpretation, where the KL term is seen as the amount of information transmitted from a sender (encoder) to a receiver (decoder) via the message ($\mathbf{z}$) (Alemi et al., 2018) and the usage of C can control this channel capacity. This can help us to make a fair comparison between Diagonal Gaussian Posterior (DGP) and IGP when VAEs are under the same encoder capacity constraint.

VAE with Isotropic Gaussian Posterior. A common behaviour of VAEs is the presence of inactive representation units across the entire dataset, causing the number of utilised dimensions to be even far smaller than the number of potential generative factors behind any real-world dataset. The soft ellipsoidal representation space of VAEs is known to lead to less representative mean vectors (Bosc and Vincent, 2020). We illustrate that encouraging isotropy (i.e., tying the variance of dimensions on the posterior) will avoid the aforementioned issue since the encoder of VAEs would be forced to either use all dimensions or none and the learned latent space is soft spherical. In the Gaussian case, this corresponds to using an Isotropic Gaussian, a subclass of diagonal Gaussian distribution $\{\mathcal{N}(\boldsymbol{\mu}, \sigma^2\boldsymbol{I})|\boldsymbol{\mu} \in \mathbf{R}^n, \sigma \in \mathbf{R}^+\}$, as the posterior. Tying the variances in IGP imposes a different pathological pattern by encouraging Active Units (AUs; Burda et al., 2016) to reach the maximum (i.e., representation dimension).

Additionally, the use of IGP allows the estimation of variance more accurately. Suppose we have N samples with the same posterior. For a K-dimension diagonal Gaussian posterior, we will have an estimation of variance with standard deviation approximately $\hat{\sigma}_k^2\sqrt{\frac{2}{N}}$ for each dimension k, whereas for an isotropic Gaussian posterior, we will have a unified estimation of variance with standard deviation approximately $\hat{\sigma}^2\sqrt{\frac{2}{NK}}$, where $\hat{\sigma}_k^2$ and $\hat{\sigma}^2$ denote the estimates of the variance. Moreover, with K different $\hat{\sigma}_k^2$ estimates, a few may differ substantially from their best values by chance.[3]

3 Experiments

We trained our models on Yahoo Question and DBpedia (Zhang et al., 2015) which have (100K/10K/10K, 12K, 10) and (140K/14K/14K, 12K, 14) for (sentences in training/dev/test, vocabulary size, classes), respectively. We use one unidirectional LSTM layer for encoder and decoder, and fully-connected layers to produce mean and variance of posteriors. We concatenate the latent

[3] It is worth noting that IGP is not a solution for posterior collapse, and our experimental findings are not specific to the chosen technique for avoiding the collapse (i.e., our preliminary experiments with KL-annealing exhibit similar findings reported in this paper).

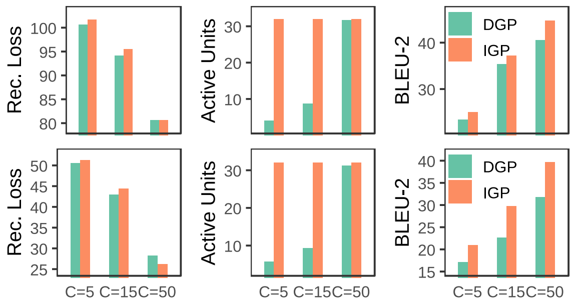


Figure 1: Results are calculated on the test set (average of 3 runs reported) of (top) DBpedia Corpus and (bottom) Yahoo Question (Zhang et al., 2015). AU is bounded by the dimensionality of z (32).

code with word embedding at every timestamp as the input of the decoder. For VAE with IGP, we just produce one variance value and assign it to be the variance of posterior for all dimensions. At decoding phase, we use greedy decoding. The dimensions for word embedding, encoder-decoder LSTMs, and latent code are (200, 512, 32). Three different values of C are used on each dataset to explore the impact of the amount of information transmitted by the code. We also adopt Autoencoder (AE) as a baseline.[4] All models are trained from 3 random starts for 20 epochs and 128 batch size using Adam (Kingma and Ba, 2015) with learning rate 0.0005.

We compare the choice of isotropic Gaussian posterior (IGP) with vanilla diagonal Gaussian posterior (DGP) on various grounds, from reconstruction loss and unit activation (§3.1) to downstream classification task, sample efficiency, robustness, and generation (§3.2), posterior sharpness (§3.3), and distributional properties of the induced representations (§3.4).

3.1 Basic Results

Figure 1 reports the reconstruction loss, AU and BLEU-2 (Papineni et al., 2002) for $C = 5, 15, 50$. KL-divergence in all cases matches the set target C. We observe the C constraint can effectively control the KL-divergence to the set level. The reconstruction loss generally drops with the increase of C. We observe the same pattern for DGP and IGP. Additionally, while DGP struggles, IGP can activate all dimensions (e.g., AU for $C = 5$ on DBpedia are 4 and 32 for DGP and IGP, respectively). This translates into IGP reaching a significantly higher BLEU. For more results, including autoencoder, see *Appendix*.

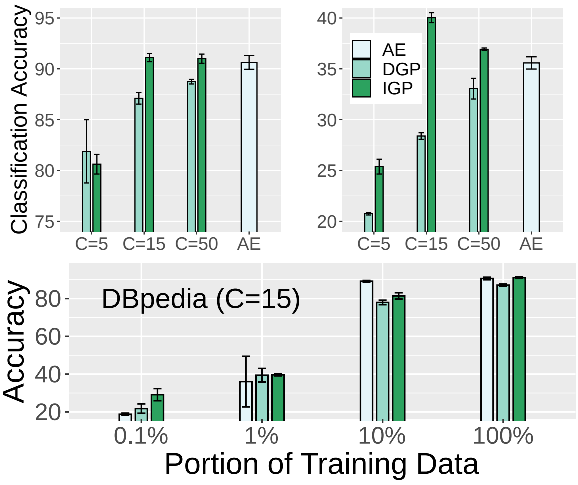


Figure 2: Classification accuracy on DBpedia (top-left) and Yahoo (top-right) with and without the isotropic Gaussian posterior (IGP) under different C values. Also, classification accuracy for $C = 15$ trained on various portion of DBpedia (bottom). Results are reported as mean and std across 3 VAE encoders.

3.2 Classification and Generation

Classification. We trained a classifier on top of the frozen encoders of DGP and IGP and use the mean vector representations as features to train the classifier. For the classifier, we used a 2-hidden-layer MLP with 128 neurons and ReLU activation function at each layer. We trained 10 randomly initialised classifiers and used the mean of classification accuracy as the final accuracy. Figure 2 (top) reports the results. Overall, the representations of most VAEs with IGP lead to a significant improvement of classification accuracy compared to vanilla VAEs. In the only exception (i.e, $C = 5$ on DBpedia), two models have comparable results with no model having any statistically significant advantage. We attribute this to having a more representative mean which is encouraged by IGP. One notable thing is that DGP does not perform as good as AEs regardless of C choice, whereas IGP ($C = 15, 50$) achieve similar and better classification accuracy on DBpedia and Yahoo Question.

We adopted few-shot setting to compare sample efficiency of both VAEs (with $C = 15$), by using 0.1%, 1% and 10% of training data of DBpedia and did classification on the test set as before. Accuracy scores are reported in Figure 2 (bottom) with IGP exhibiting a better sample efficiency. For instance,

[4] We also tried Importance Weighted Autoencoder (IWAE;Burda et al. (2016)) as another baseline commonly used in image domain. This model yields KL-collapse which is non-trivial to address given its objective function.

ORIGINAL	the carnegie library in unk washington is a building from 1911 . it was listed on the national register of historic places in 1982 .	st. marys catholic high school is a private roman catholic high school in phoenix arizona . it is located in the roman catholic diocese of phoenix .
IMPUTED	the carnegie library in unk washington $\cdots$	st. marys catholic high school is $\cdots$
DGP	the carnegie library in unk washington is a unk (unk ft) high school in the unk district of unk in the province of unk in the unk province of armenia .	st. marys catholic high school is a unk - unk school in unk unk county new jersey united states . the school is part of the unk independent school district .
IGP	the carnegie library in unk washington was built in 1909 . it was listed on the national register of historic places in unk was designed by architect john unk .	st. marys catholic high school is a private roman catholic high school in unk california . it is located in the roman catholic diocese of unk .

Table 1: Word imputation experiment.

	DBpedia	Yahoo
DGP	$[0.11, -0.63]$	$[0.10, -0.51]$
IGP	$[0.12, -6.22]$	$[0.08, -4.86]$

Table 2: Reports [$\|\|\mu\|\|_2^2$, $\log\det(\mathrm{Cov}[q_\phi(z)])$].

	DBpedia		Yahoo	
	Sample	Mean	Sample	Mean
DGP	0.72	0.62	0.72	0.63
IGP	0.76	0.77	0.78	0.76
AE	0.087		0.059	

Table 3: Isotropy score of mean and samples for DBpedia and Yahoo test sets (trained with $C = 15$).

the mean accuracy gap at 0.1% is quite significant being above 7 points, and VAE gets the gap down to 4 points at 100% (still significant).

We further investigated the robustness of the learned representations to perturbation via applying word dropout on sentences by randomly deleting 30% of words in a sentence, and repeating the classification experiment. IGP with accuracies of (83.5, 34.0) outperforms both DGP (76.4, 24.1) and AE (83.1, 30.7) on (DBpedia, Yahoo). We speculate this to be an indication of information overlap across dimensions of the representations at higher AU, offering a better recovery of information in the presence significant perturbation.

Generation. We imputed 75% of words of a sentence from the test set of DBpedia, fed it to VAE encoder and reconstructed the sentence from its latent code using IGP and DGP in Table 1. IGP successfully recovers the type of the mentioned object and completes the imputed sentence with a similar structure, whereas DGP fails to do so.

3.3 Posterior Shape

To understand the impact of isotropy on the aggregated posterior, $q_\phi(z) = \sum_{x\sim q(x)} q_\phi(z|x)$, we obtain unbiased samples of z by sampling an x from data and then $z \sim q_\phi(z|x)$, and measure the log determinant of covariance of the samples ($\log\det(\mathrm{Cov}[q_\phi(z)])$) as well as the mean of the samples to measure $||\mu||_2^2$. Table 2 reports these for $C = 15$. We observe that $\log\det(\mathrm{Cov}[q_\phi(z)])$ is significantly lower for IGP indicating a sharper approximate posteriors.

3.4 Properties of Representations

Isotropy Score. We quantitatively approximate the isotropy score (Mu and Viswanath, 2018),

$$IS(\mathcal{V}) = \frac{\min_{m\in\mathcal{M}} \sum_{v\in\mathcal{V}} \exp(m^\intercal v)}{\max_{m\in\mathcal{M}} \sum_{v\in\mathcal{V}} \exp(m^\intercal v)},$$

where $\mathcal{V}$ is the matrix of representations (i.e., of samples or mean vectors of posteriors), and $\mathcal{M}$ is the set of eigen vectors of $\mathcal{V}^\intercal\mathcal{V}$. As observed in Table 3, compared to DGP, IGP has a significantly larger IS on both means and samples. Interestingly, given that dimensions are independently modeled via univariate Gaussians, both VAEs outperform the Autoencoder counterparts.

Visualization. We visualize the learned representation space of DGP and IGP for DBpedia, using t-SNE (van der Maaten and Hinton, 2008), in Figure 3 (bottom). As illustrated in the right plot, the clusters of classes in IGP have less overlap among classes compared with DGP (left). Additionally, we use the Mapper[5] algorithm (Singh et al., 2007) to visualise the highest density region (HDR) (Hyndman, 1996) of the mean vectors for DGP and IGP. HDR cuts the overall density space to form latent spaces that contain above a threshold probability mass (i.e., ≥ 0.05 with minimum samples ≥ 2 per latent space). The output of the mapper is a graph, where each component in the graph corresponds to a set of nearby points forming a high density space.

[5] https://github.com/scikit-tda/kepler-mapper

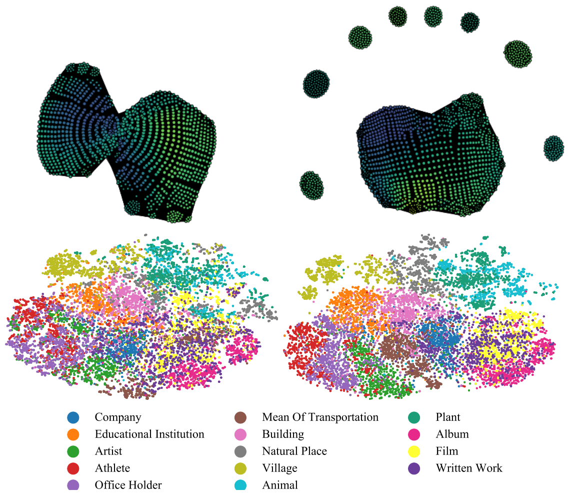


Figure 3: Visualisations of the mean representations of posterior on DBpedia test set for $C = 15$. **Left:** DGP; **Right:** IGP. **Top:** HDR; **Bottom:** t-SNE.

The connectivity of the graph reflects some topological properties of the sampling space (darker colors indicate higher density). As observed in Figure 3 (top), the HDR of DGP posterior means forms a single component whereas IGP forms 9 disconnected components indicating more discriminative characteristics of its mean vectors, echoing earlier results in better accuracy in the classification setting (§3.2).

4 Conclusion

We proposed Isotropic Gaussian Posteriors (IGP) as a means of encouraging isotropy in the latent space induced by VAEs. The injection of isotropy addressed a sub-optimal behaviour of VAEs by activating more dimensions of the representation and encouraging a more discriminative latent space. Our experiments illustrated a significant improvement of classification performance and robustness to input perturbations with IGP. We also observed, in the sentence completion task, that VAE trained with IGP is more capable at maintaining semantic cohesiveness. Our ongoing work suggests the representation utilisation achieved by IGP has the potential to be exploited towards representational properties such as disentanglement.

Acknowledgments

We thank the anonymous reviewers for their helpful suggestions.

References

Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A Saurous, and Kevin Murphy. 2018. Fixing a broken ELBO. In *International Conference on Machine Learning*, pages 159–168. PMLR.

Tom Bosc and Pascal Vincent. 2020. Do sequence-to-sequence VAEs learn global features of sentences? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4296–4318, Online. Association for Computational Linguistics.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016*, pages 10–21, Berlin, Germany. ACL.

Stephen Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge University Press.

Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. 2016. Importance weighted autoencoders. In *4th International Conference on Learning Representations, ICLR 2016, Conference Track Proceedings*, San Juan, Puerto Rico.

Jihun Choi, Taeuk Kim, and Sang-goo Lee. 2019. A cross-sentence latent variable model for semi-supervised text sequence matching. In *ACL*, pages 4747–4761.

Adji B. Dieng, Yoon Kim, Alexander M. Rush, and David M. Blei. 2019. Avoiding latent variable collapse with generative skip models. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019*, volume 89 of *Proceedings of Machine Learning Research*, pages 2397–2405, Naha, Okinawa, Japan. PMLR.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. 2019. Cyclical annealing schedule: A simple approach to mitigating KL vanishing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 240–250, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Serhii Havrylov and Ivan Titov. 2020. Preventing posterior collapse with levenshtein variational autoencoder. *CoRR*, abs/2004.14758.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. 2019. Lagging inference networks and posterior collapse in variational autoencoders. In *7th International Conference on Learning Representations, ICLR 2019*, New Orleans, LA, USA.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*, Toulon, France.

Rob J Hyndman. 1996. Computing and graphing highest density regions. *The American Statistician*, 50(2):120–126.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*, San Diego, CA, USA.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Conference Track Proceedings*, Banff, AB, Canada.

Bohan Li, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick, and Yiming Yang. 2019. A surprisingly effective fix for deep latent variable modeling of text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3603–3614, Hong Kong, China. Association for Computational Linguistics.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020a. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130, Online. Association for Computational Linguistics.

Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujun Li, Yizhe Zhang, and Jianfeng Gao. 2020b. Optimus: Organizing sentences via pre-trained modeling of a latent space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4678–4699, Online. Association for Computational Linguistics.

Fangyu Liu, Ivan Vulić, Anna Korhonen, and Nigel Collier. 2021. Fast, effective, and self-supervised: Transforming masked language models into universal lexical and sentence encoders. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1442–1459, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiaqi Mu and Pramod Viswanath. 2018. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*.

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

Victor Prokhorov, Yingzhen Li, Ehsan Shareghi, and Nigel Collier. 2021. Learning sparse sentence encoding without supervision: An exploration of sparsity in variational autoencoders. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 34–46, Online. Association for Computational Linguistics.

Victor Prokhorov, Ehsan Shareghi, Yingzhen Li, Mohammad Taher Pilehvar, and Nigel Collier. 2019. On the importance of the Kullback-Leibler divergence term in variational autoencoders for text generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation@EMNLP-IJCNLP 2019*, pages 118–127, Hong Kong. Association for Computational Linguistics.

Gurjeet Singh, Facundo Mémoli, Gunnar E Carlsson, et al. 2007. Topological methods for the analysis of high dimensional data sets and 3d object recognition. *PBG@ Eurographics*, 2.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *CoRR*, abs/2103.15316.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605.

Liangchen Wei and Zhi-Hong Deng. 2017. A variational autoencoding approach for inducing cross-lingual word embeddings. In *IJCAI*, pages 4165–4171.

Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. 2017. Variational autoencoder for semi-supervised text classification. In *AAAI*, pages 3358–3364.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3881–3890, Sydney, NSW, Australia. PMLR.

Pengcheng Yin, Chunting Zhou, Junxian He, and Graham Neubig. 2018. StructVAE: Tree-structured latent variable models for semi-supervised semantic parsing. In *ACL*, pages 754–765.

Lan Zhang, Victor Prokhorov, and Ehsan Shareghi. 2021. Unsupervised representation disentanglement of text: An evaluation on synthetic datasets. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 128–140, Online. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 649–657, Cambridge, MA, USA. MIT Press.

Yi Zhu, Ehsan Shareghi, Yingzhen Li, Roi Reichart, and Anna Korhonen. 2021. Combining deep generative models and multi-lingual pretraining for semi-supervised document classification. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 894–908, Online. Association for Computational Linguistics.

A Full Results

We report detailed reconstruction loss, KL-divergence, active units and results of BLEU and ROUGE scores on reconstructed test set in Table 4.

		Rec.	KL	AU	BLEU-2/4	ROUGE-2/4
DBpedia	AE	$66.32_{\pm 0.11}$	-	$32.0_{\pm 0.0}$	$40.96_{\pm 0.25}/27.57_{\pm 0.17}$	$35.87_{\pm 0.19}/23.78_{\pm 0.07}$
	$C = 5$, DGP	$100.65_{\pm 0.08}$	$5.09_{\pm 0.01}$	$4.0_{\pm 0.0}$	$23.47_{\pm 0.79}/14.00_{\pm 0.47}$	$20.85_{\pm 0.69}/10.19_{\pm 0.34}$
	$C = 5$, IGP	$101.73_{\pm 0.31}$	$5.04_{\pm 0.01}$	$32.0_{\pm 0.0}$	$25.09_{\pm 0.55}/14.83_{\pm 0.22}$	$22.29_{\pm 0.17}/10.47_{\pm 0.10}$
	$C = 15$, DGP	$94.16_{\pm 0.19}$	$15.06_{\pm 0.04}$	$8.7_{\pm 0.9}$	$35.35_{\pm 0.49}/22.37_{\pm 0.31}$	$30.54_{\pm 0.43}/17.41_{\pm 0.16}$
	$C = 15$, IGP	$95.52_{\pm 0.08}$	$15.08_{\pm 0.05}$	$32.0_{\pm 0.0}$	$37.23_{\pm 0.27}/24.47_{\pm 0.11}$	$34.19_{\pm 0.12}/19.32_{\pm 0.06}$
	$C = 50$, DGP	$80.65_{\pm 0.53}$	$50.02_{\pm 0.04}$	$31.7_{\pm 0.5}$	$40.54_{\pm 0.21}/26.95_{\pm 0.19}$	$35.19_{\pm 0.25}/22.13_{\pm 0.24}$
	$C = 50$, IGP	$80.58_{\pm 0.04}$	$50.15_{\pm 0.04}$	$32.0_{\pm 0.0}$	$44.79_{\pm 0.30}/30.72_{\pm 0.17}$	$39.91_{\pm 0.12}/25.40_{\pm 0.08}$
Yahoo Question	AE	$17.64_{\pm 0.28}$	-	$32.0_{\pm 0.0}$	$42.88_{\pm 0.51}/32.86_{\pm 0.58}$	$41.63_{\pm 0.58}/31.67_{\pm 0.67}$
	$C = 5$, DGP	$50.58_{\pm 0.06}$	$5.14_{\pm 0.01}$	$5.7_{\pm 0.5}$	$17.07_{\pm 0.71}/6.04_{\pm 0.25}$	$10.96_{\pm 0.41}/1.50_{\pm 0.06}$
	$C = 5$, IGP	$51.24_{\pm 0.01}$	$5.06_{\pm 0.03}$	$32.0_{\pm 0.0}$	$20.91_{\pm 0.03}/8.07_{\pm 0.03}$	$14.48_{\pm 0.13}/2.21_{\pm 0.02}$
	$C = 15$, DGP	$43.00_{\pm 0.12}$	$15.06_{\pm 0.04}$	$9.3_{\pm 1.2}$	$22.62_{\pm 0.37}/10.81_{\pm 0.21}$	$16.04_{\pm 0.32}/4.76_{\pm 0.08}$
	$C = 15$, IGP	$44.43_{\pm 0.05}$	$15.20_{\pm 0.12}$	$32.0_{\pm 0.0}$	$29.76_{\pm 0.06}/14.99_{\pm 0.08}$	$23.11_{\pm 0.17}/6.94_{\pm 0.12}$
	$C = 50$, DGP	$28.29_{\pm 0.40}$	$50.00_{\pm 0.19}$	$31.3_{\pm 0.9}$	$31.78_{\pm 0.73}/20.47_{\pm 0.70}$	$27.14_{\pm 0.85}/15.07_{\pm 0.77}$
	$C = 50$, IGP	$26.18_{\pm 0.19}$	$50.15_{\pm 0.08}$	$32.0_{\pm 0.0}$	$39.68_{\pm 0.20}/27.49_{\pm 0.31}$	$35.73_{\pm 0.40}/22.57_{\pm 0.55}$

Table 4: Results are calculated on the test set. We report mean value and standard deviation across 3 runs. Rec and AU denote reconstruction loss and number of Active Units, respectively. DGP, and IGP denote diagonal Gaussian posteriors and isotropic Gaussian posteriors, respectively. C is the target KL value.

Efficient Classification of Long Documents Using Transformers

Hyunji Hayley Park
University of Illinois*
hpark129@illinois.edu

Yogarshi Vyas
AWS AI Labs
yogarshi@amazon.com

Kashif Shah
Microsoft*
kashifshah@microsoft.com

Abstract

Several methods have been proposed for classifying long textual documents using Transformers. However, there is a lack of consensus on a benchmark to enable a fair comparison among different approaches. In this paper, we provide a comprehensive evaluation of the relative efficacy measured against various baselines and diverse datasets — both in terms of accuracy as well as time and space overheads. Our datasets cover binary, multi-class, and multi-label classification tasks and represent various ways information is organized in a long text (e.g. information that is critical to making the classification decision is at the beginning or toward the end of the document). Our results show that more complex models often fail to outperform simple baselines and yield inconsistent performance across datasets. These findings emphasize the need for future studies to consider comprehensive baselines and datasets that better represent the task of long document classification to develop robust models.[1]

1 Introduction

Transformer-based models (Vaswani et al., 2017) have achieved much progress across many areas of NLP including text classification (Minaee et al., 2021). However, such progress is often limited to short sequences because self-attention requires quadratic computational time and space with respect to the input sequence length. Widely-used models like BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019) are typically pretrained to process up to 512 tokens. This is problematic because real-world data can be arbitrarily long. As such, different models and strategies have been proposed to process longer sequences.

In particular, we can identify a few standard approaches for the task of long document classification. The simplest approach is to truncate long documents — using BERT or RoBERTa on the first 512 tokens is often used as a baseline. More efficient Transformer models like Longformer (Beltagy et al., 2020) and Big Bird (Zaheer et al., 2020) use sparse self-attention instead of full self-attention to process longer documents (e.g. up to 4,096 tokens). Other approaches process long documents in their entirety by dividing them into smaller chunks (e.g. Pappagari et al., 2019). An alternative idea proposed by recent work is to select sentences from the document that are salient to making the classification decision (Ding et al., 2020).

However, the relative efficacy of these models is not very clear due to a lack of consensus on benchmark datasets and baselines. Tay et al. (2021) propose a benchmark for comparing Transformers that can operate over long sequences, but this only includes a single, simulated[2] long document classification task. Novel variants of efficient Transformers are often compared to a BERT/RoBERTa baseline only, without much comparison to other Transformer models designed for the task (e.g. Beltagy et al., 2020; Zaheer et al., 2020). Conversely, models designed for long document classification often focus exclusively on state-of-the-art models for particular datasets, and do not consider a BERT/RoBERTa baseline or any other Transformer models (e.g. Ding et al., 2020; Pappagari et al., 2019).

This paper provides a much-needed comprehensive comparison among existing models for long document classification by evaluating them against unified datasets and baselines. We compare models that represent different approaches on various datasets and against Transformer baselines. Our datasets cover binary, multi-class, and multi-label

*Work done while at Amazon

[1] Our code is available at https://github.com/amazon-research/efficient-longdoc-classification.

[2] The benchmark considers the task of classifying IMDB reviews (Maas et al., 2011) using byte-level information to simulate longer documents.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 702 - 709
May 22-27, 2022 ©2022 Association for Computational Linguistics

classification. We also consider different ways information that is relevant to the classification is organized in texts (e.g. in the beginning or toward the end) and how this affects model performance. We also compare the models in terms of their training time, inference time, and GPU memory requirements to account for additional complexity that some of the models have relative to a BERT baseline. This allows us to compare the practical efficacy of the models for real-world usage.

Our results show that more sophisticated models are often outperformed by simpler models (often including a BERT baseline) and yield inconsistent performance across datasets. Based on these findings, we highlight the importance of considering diverse datasets while developing models, especially those that represent different ways key information is presented in long texts. Additionally, we recommend that future research should also always include simpler baseline models. To summarize, our contributions are:

- We provide insights into the practical efficacy of existing models for long document classification by evaluating them across different datasets, and against several baselines. We compare the accuracy of these models as well as their runtime and memory requirements.
- We present a comprehensive suite of evaluation datasets for long document classification with various data settings for future studies.
- We propose simple models that often outperform complex models and can be challenging baselines for future models for this task.

2 Methods

In this paper, we compare models representing different approaches to long document classification (Beltagy et al., 2020; Pappagari et al., 2019; Ding et al., 2020) on unified datasets and baselines.

2.1 Existing Models

As described in §1, four distinct approaches have been proposed for long document classification: 1) document truncation, 2) efficient self-attention, 3) chunk representations, 4) key sentence selection. We evaluate a representative model from each category in this work.

BERT (document truncation) The simplest approach consists of finetuning BERT after truncating long documents to the first 512 tokens.[3] As in Devlin et al. (2019), we use a fully-connected layer on the [CLS] token for classification. This is an essential baseline as it establishes the limitations of a vanilla BERT model in classifying long documents yet is still competitive (e.g. Beltagy et al., 2020; Chalkidis et al., 2019). However, some prior work fails to consider this baseline (e.g. Ding et al., 2020; Pappagari et al., 2019).

Longformer (efficient self-attention) We select Longformer (Beltagy et al., 2020) as a model designed to process longer input sequences based on efficient self-attention that scales linearly with the length of the input sequence (see Tay et al., 2020, for a detailed survey). Longformer also truncates the input, but it has the capacity to process up to 4,096 tokens rather than 512 tokens as in BERT. Following Beltagy et al. (2020), we use a fully-connected layer on top of the first [CLS] token with global attention. Longformer outperformed a RoBERTa baseline significantly on a small binary classification dataset (Beltagy et al., 2020). However, it has not been evaluated against any other models for text classification or on larger datasets that contain long documents.

ToBERT (chunk representations) Transformer over BERT (ToBERT, Pappagari et al., 2019) takes a hierarchical approach that can process documents of any lengths in their entirety. The model divides long documents into smaller chunks of 200 tokens and uses a Transformer layer over BERT-based chunk representations. It is reported to outperform previous state-of-the-art models on datasets of spoken conversations. However, it has not been compared to other Transformer models. We reimplement this model based on the specifications reported in Pappagari et al. (2019) as the code is not publicly available.

CogLTX (key sentence selection) Cognize Long TeXts (CogLTX, Ding et al., 2020) jointly trains two BERT (or RoBERTa) models to select key sentences from long documents for various tasks including text classification. The underlying idea that a few key sentences are sufficient for a given task has been explored for question answering (e.g. Min et al., 2018), but not much for text classification. It is reported to outperform ToBERT and some other

[3] In practice, the first 510 tokens are used along with the [CLS] and [SEP] tokens. We use the token count including the two special tokens throughout the paper for simplicity.

neural models (e.g. CNN), but it is not evaluated against other Transformer models.

We use their multi-class classification code for any classification task with appropriate loss functions.[4] Following Beltagy et al. (2020), we use sigmoid and binary cross entropy loss on the logit output of the models for binary classification. The same setting is used for multi-label classification with softmax normalization and cross entropy loss.

2.2 Novel Baselines

In addition to the representative models above, we include two novel methods that serve as simple but strong baseline models.

BERT+TextRank While the BERT truncation baseline is often effective, key information required to classify documents is not always found within the first 512 tokens. To account for this, we augment the first 512 tokens, with a second set of 512 tokens obtained via TextRank, an efficient unsupervised sentence ranking algorithm (Mihalcea and Tarau, 2004). TextRank provides an efficient alternative to more complex models designed to select key sentences such as CogLTX. Specifically, we concatenate the BERT representation of the first 512 tokens with that of the top ranked sentences from TextRank (up to another 512 tokens). As before, we use a fully-connected layer on top of the concatenated representation for classification. We use PyTextRank (Nathan, 2016) as part of the spaCy pipeline (Honnibal et al., 2020) for the implementation with the default settings.

BERT+Random As an alternative approach to the BERT+TextRank model, we select random sentences up to 512 tokens to augment the first 512 tokens. Like BERT+TextRank, this can be a simple baseline approach in case key information is missing in truncated documents.[5]

2.3 Hyperparameters

We use reported hyperparameters for the existing models whenever available. However, given that we include different datasets that the original papers did not use, we additionally explore different hyperparameters for the models. Detailed information is available in Appendix A.

Dataset	# BERT Tokens	% Long
Hyperpartisan	744.2 ± 677.9	53.5
20NewsGroups	368.8 ± 783.8	14.7
EURLEX-57K	707.99 ± 538.7	51.3
Book Summary	574.3 ± 659.6	38.8
– Paired	1,148.6 ± 933.9	75.5

Table 1: Statistics on the datasets. # BERT Tokens refers to the average token count obtained via the tokenizer of the BERT base (uncased) model. % Long refers to the percentage of documents with over 512 BERT tokens.

2.4 Data

We select three classification datasets containing long documents to cover various kinds of classification tasks: Hyperpartisan (Kiesel et al., 2019) (binary classification), 20NewsGroups (Lang, 1995) (multi-class classification) and EURLEX-57K (Chalkidis et al., 2019) (multi-label classification). We also re-purpose the CMU Book Summary Dataset (Bamman and Smith, 2013) as an additional multi-label classification dataset.

We also modify the EURLEX and Book Summary datasets to represent different data settings and further test all models under these challenging variations. A document in the EURLEX dataset contains a legal text divided into several sections, and the first two sections (header, recitals) carry the most relevant information for classification (Chalkidis et al., 2019). We invert the order of the sections so that this key information is located toward the end of each document (Inverted EURLEX). This creates a dataset particularly challenging for models that focus only on the first 512 tokens. We also combine pairs of book summaries from the CMU Book Summary dataset to create a new dataset (Paired Book Summary) that contains longer documents with two distinctive information blocks. Again, this challenges models not to solely rely on the signals from the first 512 tokens. In addition, it further challenges models to detect two separate sets of signals for correct classification results. In all, these modified datasets represent different ways information may be presented in long texts and test how robust the existing models are to these. Table 1 summarizes characteristics of all our datasets, with more details in Appendix B.

2.5 Metrics

For the binary (Hyperpartisan) and multi-class (20NewsGroups) classification tasks, we report ac-

[4] `https://github.com/Sleepychord/CogLTX`

[5] For simplicity, sentences included in the first 512 tokens are not excluded in the random selection process. Different settings are possible, but our preliminary results did not show much difference.

Model	Hyper- partisan	20News Groups	EURLEX	Inverted EURLEX	Book Summary	Paired Summary
BERT	92.00	84.79	73.09	70.53	58.18	52.24
BERT+TextRank	91.15	84.99	72.87	71.30	58.94	55.99
BERT+Random	89.23	84.65	**73.22**	**71.47**	**59.36**	56.58
Longformer	**95.69**	83.39	54.53	56.47	56.53	**57.76**
ToBERT	89.54	**85.52**	67.57	67.31	58.16	57.08
CogLTX	94.77	84.63	70.13	70.80	58.27	55.91

Table 2: Performance metrics on the test set for all datasets. The average accuracy (%) over five runs is reported for Hyperpartisan and 20NewsGroups while the average micro-F_1 (%) is used for the other datasets. The highest value per column is in bold and the second highest value is underlined. Results below the BERT baseline are shaded.

curacy (%) on the test set. For the rest, multi-label classification datasets, we use micro-F_1 (%), which is based on summing up the individual true positives, false positives, and false negatives for each class.[6]

Model	Train Time	Inference Time	GPU Memory
BERT	1.00	1.00	<16
+TextRank	1.96	1.96	16
+Random	1.98	2.00	16
Longformer	12.05	11.92	32
ToBERT	1.19	1.70	32
CogLTX	104.52	12.53	<16

Table 3: Runtime and memory requirements of each model, relative to BERT, based on experiments on the Hyperpartisan dataset. Training and inference time were measured and compared in seconds per epoch. GPU memory requirement is in GB. Longformer and ToBERT were trained on a GPU with a larger memory and compared to a comparable run on the machine.

3 Results

Table 2 summarizes the average performance of the models over five runs with different random seeds. Overall, the key takeaway is that more sophisticated models (Longformer, ToBERT, CogLTX) do not outperform the baseline models across the board. In fact, these models are significantly more accurate than the baselines only on two datasets. As reported in Beltagy et al. (2020), Longformer recorded the strongest performance on Hyperpartisan, with CogLTX also performing well. Longformer and ToBERT performed the best for Paired Book Summary. Paired Book Summary seems to be most challenging for all models across the board and is the only dataset where the BERT baseline did the worst. However, it is worth noting that simple augmentations of the BERT baseline as in BERT+TextRank and BERT+Random were not far behind the best performing model even for this challenging dataset. ToBERT's reported performance was the highest for 20NewsGroups, but we were unable to reproduce the results due to its memory constraints. For the other datasets, these more sophisticated models were outperformed by the baselines. In particular, the simplest BERT baseline that truncates documents up to the first 512 tokens shows competitive performance overall, outperforming the majority of models for Hyperpartisan, 20NewsGroups and EURLEX. It is only the Paired Book Summary dataset where the BERT baseline performed particularly worse than other models. In general, we observe little-to-no performance gains from more sophisticated models across the datasets as compared to simpler models. A similar trend was observed even when the models were evaluated only on long documents in the test set (Appendix C). These finding suggests that the existing models do not necessarily work better for long documents across the board when diverse datasets are considered.

The relatively inconsistent performance of these existing models is even more underwhelming considering the difference in runtime and memory requirements as summarized in Table 3. Compared to BERT on the first 512 tokens, Longformer takes about 12x more time for training and inference while CogLTX takes even longer. ToBERT is faster than those two, but it requires much more GPU memory to process long documents in their entirety. Taken together with the inconsistency in

[6] The choice of these metrics are based on previous literature. An exploration of other metrics (e.g. macro-F_1) may provide further insights. However, we did not see significant differences in preliminary results, and we believe the general trend of results would not differ.

accuracy/F1 scores, this suggests that sophisticated models are not necessarily a good fit for real word use cases where efficiency is critical.

4 Discussion and Recommendations

Our results show that complex models for long document classification do not consistently outperform simple baselines. The fact that the existing models were often outperformed by the simplest BERT baseline suggests that the datasets tend to have key information accessible in the first 512 tokens. This is somewhat expected as the first two sections of EURLEX are reported to carry the most information (Chalkidis et al., 2019) and 20NewsGroups contains mostly short documents. Including these datasets to evaluate models for long document classification is still reasonable given that a good model should work well across different settings. However, these datasets alone do not represent various ways information is presented in long texts.

Instead, future studies should evaluate their models across various datasets to create robust models. While it is often difficult to obtain datasets suited for long document classification, our modifications of existing datasets may provide ways to repurpose existing datasets for future studies. We invert the order of the sections of EURLEX to create the Inverted EURLEX dataset, where key information is likely to appear toward the end of each document. Our results in Table 2 show that selective models (BERT+TextRank, BERT+Random, CogLTX) performed better than those that read longer consecutive sequences (Longformer, ToBERT) on this dataset. This suggests that this inverted dataset may contain parts of texts that should be ignored for better performance, thus providing a novel test bed for future studies. The Paired Book Summary dataset presents another challenging data setting with two distinctive information blocks. While Longformer and ToBERT performed significantly better for this dataset than others, the overall model performance was quite underwhelming, leaving room for improvement for future models.

Many of these findings were revealed only due to the choice of relevant baselines, and future work will benefit from including these as well. A BERT/RoBERTa baseline is essential to motivate the problem of long document classification using Transformers and reveal how much information is retrievable in the first 512 tokens. BERT+TextRank and BERT+Random are stronger baselines that often outperform more complex models that select key sentences. In fact, they outperformed CogLTX on five of the six datasets.

5 Conclusion

Several approaches have been proposed to use Transformers to classify long documents, yet their relative efficacy remains unknown. In this paper, we compare existing models and baselines on various datasets and in terms of their time and space requirements. Our results show that existing models, while requiring more time and/or space, do not perform consistently well across datasets, and are often outperformed by baseline models. Future studies should consider the baselines and datasets to establish robust performance.

Acknowledgments

We would like to thank the reviewers and area chairs for their thoughtful comments and suggestions. We also thank the members of AWS AI Labs for many useful discussions and feedback that shaped this work.

References

David Bamman and Noah A. Smith. 2013. New alignment methods for discriminative book summarization. *arXiv:1305.1319*.

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

Ilias Chalkidis, Emmanouil Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. 2019. Large-scale multi-label text classification on EU legislation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6314–6322, Florence, Italy. Association for Computational Linguistics.

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

Ming Ding, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. CogLTX: Applying BERT to long texts. In *Advances in Neural Information Processing Systems*, volume 33, pages 12792–12804. Curran Associates, Inc.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength natural language processing in Python.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv:1907.11692*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1725–1735, Melbourne, Australia. Association for Computational Linguistics.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning–based text classification. *ACM Computing Surveys*, 54(3):1–40.

Paco Nathan. 2016. PyTextRank, a Python implementation of TextRank for phrase extraction and summarization of text documents.

Raghavendra Pappagari, Piotr Zelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. 2019. Hierarchical Transformers for long document classification. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 838–844.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2021. Long range arena: A benchmark for efficient Transformers. In *International Conference on Learning Representations*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient Transformers: A survey. *arXiv:2009.06732*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008. Curran Associates, Inc.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. *arXiv:2007.14062*.

A Hyperparameters

Across all datasets, we used Adam optimizer with a learning rate of {5e-5, 3e-5, 0.005} for one run of each model and picked the best performing learning rate for the model. The learning rate of 0.005 was used for Longformer only because it did not perform well with a learning rate of 5e-5 or 3e-5 for most of the datasets. We set dropout rate at 0.1 as suggested by Devlin et al. (2019). The number of epochs needed for finetuning the models for different datasets is likely to vary, so we trained all models for 20 epochs and selected the best performing model based on the performance metric on the validation set. We report the average results on the test set over five different seeds.

All experiments on baseline models and CogLTX were conducted on a single Tesla V100 GPU with 16GB memory. For Longformer and ToBERT, we used a NVIDIA A100 SXM4 with 40GB memory. More details on the selected hyperparameters are available with our code at `https://github.com/amazon-research/efficient-longdoc-classification`.

B Datasets

Hyperpartisan is a binary classification dataset, where each article is labeled as True (hyperpartisan) or False (not hyperpartisan) (Kiesel et al., 2019). More than half of the documents exceed 512 tokens. It is quite different from other datasets in that it is a very small dataset: the training set contains 516 documents while the development and test sets contain 64 and 65 documents, respectively.

20NewsGroups is a widely-used multi-class classification dataset (Lang, 1995). The documents are categorized into well-balanced, 20 classes. Only

Dataset	Type	# Train	# Dev	# Test	# Labels	# BERT Tokens	% Long
Hyperpartisan	binary	516	64	65	2	744.18 ± 677.87	53.49
20NewsGroups	multi-class	10,182	1,132	7,532	20	368.83 ± 783.84	14.71
EURLEX-57K	multi-label	45,000	6,000	6,000	4,271	707.99 ± 538.69	51.30
– Inverted							
Book Summary	multi-label	10,230	1,279	1,279	227	574.31 ± 659.56	38.76
– Paired	multi-label	5,115	639	639	227	1,148.62 ± 933.97	75.54

Table 4: Statistics on the datasets. # BERT Tokens refers to the average token count obtained via the tokenizer of the BERT base model (uncased). % Long refers to the percentage of documents with more than 512 BERT tokens.

Model	Hyperpartisan	20News Groups	EURLEX	Inverted EURLEX	Book Summary	Paired Summary
BERT	88.00	86.09	66.76	62.88	60.56	52.23
BERT+TextRank	85.63	85.55	66.56	64.22	61.76	56.24
BERT+Random	83.50	**86.18**	**67.03**	**64.31**	**62.34**	56.77
Longformer	**93.17**	85.50	44.66	47.00	59.66	**58.85**
ToBERT	86.50	–	61.85	59.50	61.38	58.17
CogLTX	91.91	86.07	61.95	63.00	60.71	55.74

Table 5: Performance metrics evaluated on long documents in the test set for all datasets. The average accuracy (%) over five runs is reported for Hyperpartisan and 20NewsGroups while the average micro-F_1 (%) is used for the other datasets. The highest value per column is in bold and the second highest value is underlined. Results below the BERT baseline are shaded. Running ToBERT on 20NewsGroups seems to require further preprocessing, which we were unable to replicate with the reported information.

about 15% of the documents exceed 512 tokens. While the original dataset comes in train and test sets only, we report results on the train/dev/test split as used in Pappagari et al. (2019), where we take 10% of the original train set as the development set. Note that CogLTX reported their accuracy at 87.00% on the test set and 87.40% on the long documents in the test set, using the original train and test sets only. Our implementation of CogLTX in the same setting with five different runs resulted in a much lower performance at 85.15% on the test set and 86.57% on the long documents only. In addition, we were unable to replicate ToBERT results on 20NewsGroups. It is unclear how the dataset is further preporcessed for ToBERT, and our implementation of ToBERT caused a GPU out-of-memory error on 20NewsGroups. Thus, we show the reported results for ToBERT on this dataset.

EURLEX-57K is a multi-label classification dataset based on EU legal documents (Chalkidis et al., 2019). In total, there are 4,271 labels available, and some of them do not appear in the training set often or at all, making it a very challenging dataset. About half of the datasets are long documents. Each document contains four major zones: header, recitals, main body, and attachments. Chalkidis et al. (2019) observe that processing the first two sections only (header and recitals) results in almost the same performance as the full documents and that BERT on the first 512 tokens outperforms all the other models they considered. After examining the dataset, we exclude the attachments section as it does not seem to provide much textual information.

CMU Book Summary contains book summaries extracted from Wikipedia with corresponding metadata from Freebase such as the book author and genre (Bamman and Smith, 2013). We use the summaries and their corresponding genres for a multi-label classification task. We keep 12,788 out of 16,559 documents after removing data points missing any genre information and/or adequate summary information (e.g. less than 10 words). In total, there are 227 genre labels such as ‘Fiction’ and ‘Children’s literature’.

C Results on long documents only

Table 5 shows the results as evaluated on long documents (with over 512 tokens) in the test set only. Overall, the results show a similar trend as observed in Table 2, which reports the results on the entire documents in the test set. In general, the existing models were often outperformed by the BERT truncation baseline. This suggest that these models designed for long document classification do not perform particularly well on the long documents in the datasets. The only difference is that BERT+Random and ToBERT perform better than the BERT baseline when evaluated on long documents only for 20NewsGroups and Book Summary, respectively. However, the performance gain does not seem significant, and the relative performance with respect to the other models remains largely unchanged. In general, the relative strength of a model for a given dataset stays the same whether or not the model is evaluated on the entire documents or long documents in the test set.

Rewarding Semantic Similarity under Optimized Alignments for AMR-to-Text Generation

Lisa Jin and **Daniel Gildea**
Department of Computer Science
University of Rochester
Rochester, NY 14627

Abstract

A common way to combat exposure bias is by applying scores from evaluation metrics as rewards in reinforcement learning (RL). Metrics leveraging contextualized embeddings appear more flexible than those that match n-grams and thus ideal as training rewards. Yet metrics such as BERTSCORE greedily align candidate and reference tokens, which can give system outputs excess credit relative to a reference. Past systems using such semantic similarity rewards further suffer from repetitive outputs and overfitting. To address these issues, we propose metrics that replace the greedy alignments in BERTSCORE with optimized ones. Our model optimizing discrete alignment metrics consistently outperforms cross-entropy and BLEU reward baselines on AMR-to-text generation. Additionally, we find that this model enjoys stable training relative to a non-RL setting.

1 Introduction

Automatic evaluation metrics often score natural language generation (NLG) system outputs based on how well they lexically align to human-annotated references. In tasks such as machine translation and summarization, these metrics may unfairly penalize outputs that express the correct semantics despite a lower n-gram overlap with reference strings. As a result, models overfitting to certain token-level patterns may dominate those generating more creatively (e.g., through synonyms or varied sentence structure).

NLG systems are typically trained to maximize likelihood of a single set of references. Conditioning models on gold prefixes shields them from their own predictions during training—an issue known as exposure bias. Adding reinforcement learning (RL) objectives (Ranzato et al., 2016; Edunov et al., 2018) can aid exploration by giving a model feedback on sequences sampled from its own distribution. However, it is common practice to use automatic evaluation scores like BLEU (Papineni et al., 2002) and ROUGE (Lin and Hovy, 2002) as sequence-level rewards. This results in the same lack of semantic signal described earlier.

Instead of hinging evaluation on hard n-gram overlap, recent metrics (Zhang et al., 2019; Zhao et al., 2019) rely on vector similarity between contextualized subword embeddings to make more semantically faithful judgments. BERTSCORE, in particular F_{BERT}, computes a token-level F1 score based on greedy alignment of similar embeddings. With their strength in offline evaluation, it is natural to ask how these embeddings-based metrics can help provide more realistic training feedback.

Past approaches to train models with semantic similarity scores include both non-differentiable and differentiable objectives. Wieting et al. (2019) separately train paraphrastic sentence embeddings that provide semantic similarity rewards to a neural machine translation (NMT) system. Rewards were included in a mixed minimum risk and maximum likelihood training phase. Besides an embedding training overhead, the model needed a length penalty term to limit repetitive outputs. Li et al. (2019) adopt a similar fine-tuning approach using an RL objective with F_{BERT} for abstractive summarization. While their models were less repetitive, their news domain corpora may have been a natural match for BERT embeddings. Finally, Jauregi Unanue et al. (2021) also propose to optimize F_{BERT} but with fully differentiable training objectives in NMT. Yet their models overfit after only a few epochs and scored lower in BLEU at the cost of higher F_{BERT}. We hypothesize that metrics employing external pretrained vectors may suffer from domain mismatch with downstream data. This can hurt the accuracy of semantic similarity scores computed during training.

In this work, we focus on text generation from Abstract Meaning Representations (AMRs, Banarescu et al., 2013), sentence-level semantic graphs that are rooted, directed, and acyclic. This

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 710 - 715
May 22-27, 2022 ©2022 Association for Computational Linguistics

task's models may especially benefit from an emphasis on semantic rather than lexical similarity. It also provides a challenging setting to evaluate overfitting given the relatively small corpus size.

In our analysis of F_{BERT} rewards, we note that F_{BERT} could worsen repetition and incomplete outputs in NLG systems. Due to its greedy token alignment, F_{BERT} precision may assign extra credit to a reference token 'retrieved' multiple times. In response, we contribute the following.

- We introduce metrics that apply discrete and continuous alignments to BERTSCORE, mitigating the pitfalls of greedy alignment.
- For text generation from AMR, we are the first to train on RL objectives with embeddings-based evaluation metrics.
- As RL rewards, we compute BERTSCORE-based metrics on a model's own token representations rather than BERT embeddings. This is more memory-efficient and does not overfit relative to pure cross-entropy training.

2 Greedy Token Alignment

The main insight behind BERTSCORE and related metrics is to align hypothesis and reference tokens using their pairwise vector similarity scores. These alignments are later used to weight the contribution of token-level similarity scores towards a final sequence-level score. Concretely, given $(\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_m)$ and $(\mathbf{y}_1, \ldots, \mathbf{y}_k)$ hypothesis and reference token embeddings, precision in F_{BERT} is

$$P_{\text{BERT}} = \frac{1}{m} \sum_{\hat{y}_i \in \hat{y}} \max_{y_j \in y} \cos(\hat{\mathbf{y}}_i, \mathbf{y}_j),$$

where $\cos(\hat{\mathbf{y}}, \mathbf{y}) = \hat{\mathbf{y}}^\top \mathbf{y} / \|\hat{\mathbf{y}}\| \, \|\mathbf{y}\|$ denotes cosine similarity. Each hypothesis token $\hat{y}_i$ is greedily aligned to the reference token y_j with the highest corresponding embedding cosine similarity. Unlike in BLEU, P_{BERT} does not clip the number of times $\hat{y}_i$ can align to a unique y_j by its count in y. As such, a hypothesis will get excess credit by repeating a reference token beyond this count. While the authors claim greedy alignments have little effect on BERTSCORE evaluation performance, they perform poorly relative to metrics based on optimized alignments in our experiments.

3 Optimized Token Alignment

Aligning tokens between hypothesis and reference can be seen as an assignment problem, where a token pair $(\hat{y}_i, y_j)$ is highly weighted if it incurs low cost (i.e., distance).

Here, we describe discrete token matching (one-to-one) and soft alignment (one-to-many). For the latter, we extract alignments from the earth mover's distance (EMD, Villani, 2009; Peyré and Cuturi, 2019) transport matrix. We weight pairwise token similarities as in F_{BERT} using each of these two alignments to provide metrics F_{DISC} and F_{CONT}.

3.1 Discrete word matching

To avoid the issues with greedy alignment in P_{BERT}, we can extract one-to-one alignments between the two sequences. Let $C \in \mathbb{R}^{m \times k}$ denote the pairwise cosine distance matrix such that $C_{ij} = 1 - \cos(\hat{\mathbf{y}}_i, \mathbf{y}_j)$. For notational clarity, let $\widetilde{C} = 1 - C$. We wish to find alignments

$$T^d = \operatorname*{arg\,min}_{T \in \{0,1\}^{m \times k}} \sum_{i=1}^{m} \sum_{j=1}^{k} T_{ij} C_{ij}, \qquad (1)$$

such that no element in $\mathbf{h} = T\mathbf{1}_k$ and $\mathbf{r} = T^\top \mathbf{1}_m$ exceeds one. In other words, each $\hat{y}_i$ can align to at most one y_j (exactly one when $m = k$), and vice versa. This linear sum assignment problem can be solved in low-order polynomial time (Crouse, 2016), making it suitable for use during training.

Metric The updated precision is found as

$$P_{\text{DISC}} = \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{k} T^d_{ij} \widetilde{C}_{ij}. \qquad (2)$$

Recall R_{DISC} takes an analogous form and is combined with P_{DISC} to produce an F1 score, F_{DISC}.

3.2 Continuous word alignment

We also experiment with soft alignments, where weights in T are continuous. In the case of P_{BERT}, one-to-many alignments between each hypothesis token $\hat{y}_i$ and those in $\{y_j\}_{j \in [k]}$ are permitted.

Inspired by work applying EMD to semantic text similarity (Kusner et al., 2015; Clark et al., 2019), we frame alignment as minimizing the transportation cost between token embeddings from the hypothesis and reference distributions. The amount of token-level mass to transport between the two distributions is $\mathbf{h}$ and $\mathbf{r}$, respectively. Instead of assigning IDF as the mass per token (Zhao et al., 2019), we use the norm of its embedding (i.e., $\|\mathbf{y}\|$, Yokoi et al., 2020) for simplicity.

The EMD, or optimal transport, problem is

$$T^c = \underset{T \in \mathbb{R}_{\geq 0}^{m \times k}}{\arg\min} \sum_{i=1}^{m} \sum_{j=1}^{k} T_{ij} C_{ij}, \tag{3}$$
$$\text{s.t.} \quad \mathbf{h} = T\mathbf{1}_k, \; \mathbf{r} = T^\top \mathbf{1}_m.$$

Intuitively, if we view T_{ij} as the joint probability of aligning $\hat{y}_i$ with y_j, the row and column sums are marginals (Cuturi, 2013).

Metric To compute F_{CONT}, we normalize the alignment weights such that the rows of T sum to one for precision, and the columns for recall.

$$P_{\text{CONT}} = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{h_i} \sum_{j=1}^{k} T_{ij}^c \widetilde{C}_{ij}, \tag{4}$$

$$R_{\text{CONT}} = \frac{1}{k} \sum_{j=1}^{k} \frac{1}{r_j} \sum_{i=1}^{m} T_{ij}^c \widetilde{C}_{ij} \tag{5}$$

4 Semantic Similarity Rewards

We propose to fine-tune on our optimized F1 metrics, applying a weighted average of cross-entropy and RL objectives. Given source sequence x (e.g., a linearized AMR), the former is computed as

$$\mathcal{L}_e = -\sum_{i=1}^{k} \log p(y_i \mid y_{<i}, x).$$

To encourage close evaluation scores between sampled $\bar{y}$ and reference y, the RL objective is

$$\mathcal{L}_r = (\Delta(\bar{y}_g, y) - \Delta(\bar{y}, y)) \sum_{i=1}^{k} \log p(\bar{y}_i \mid \bar{y}_{<i}, x),$$

where Δ is the chosen evaluation metric and $\bar{y}_g$ is a greedily decoded baseline relative to $\bar{y}$. This baseline helps reduce variance in REINFORCE (Williams, 1992). The combined cross-entropy and RL loss is

$$\mathcal{L} = \lambda \mathcal{L}_r + (1 - \lambda)\mathcal{L}_e,$$

where λ is empirically set to 0.3.

5 Experiments

We examine the performance of our proposed metrics as RL rewards on AMR-to-text generation.

	BLEU	METEOR	CHRF	BLEURT
XENT	36.37	39.94	65.68	56.30
BL-R	37.06	40.30	66.19	56.08
F_{BERT}	36.06	39.85	65.23	55.45
F_{CONT}	36.91	40.34	66.07	55.96
F_{DISC}	**37.65**	**40.61**	**66.55**	**57.01**

Table 1: Results on the AMR2017T10 test set.

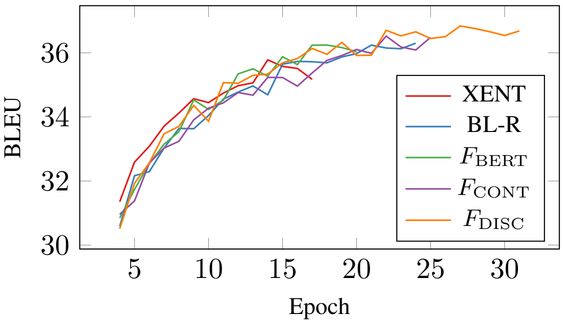


Figure 1: Development set BLEU during fine-tuning.

5.1 Setup

Dataset The LDC2017T10 dataset that we experiment on contains ~36K training and ~1.4K each of development and test AMR-sentence pairs. To leverage strong pre-trained language models, the AMRs are linearized as in Ribeiro et al. (2021).

Evaluation We report results in terms of BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), CHRF (Popović, 2015), and BLEURT (Sellam et al., 2020). Only the latter metric makes use of pre-trained contextualized embeddings.

Baselines For all experiments, we fine-tune the small capacity T5 model (Raffel et al., 2020) from Ribeiro et al. (2021). The model has 60M parameters and features a Transformer-based encoder and decoder. We compare our F_{DISC} and F_{CONT} metrics for RL-based training against three baseline approaches. XENT is a pure cross-entropy objective. For RL-based approaches, we include a BLEU reward (BL-R) and one with F_{BERT}—computed on the lowest level token embeddings in T5.[1] The λ scaling factor for the RL objective is set to 0.3 across all RL-based experiments.

Implementation details *Adam* (Kingma and Ba, 2015) is used to optimize the model with an initial

[1] This also applies to F_{DISC} and F_{CONT}.

(1)	REF	There are **12 teams totally participating** in the competition.
	XENT	The competition was **part of a total of 12 teams**.
	F_{BERT}	The competition is **part of a total of 12 teams**.
	F_{DISC}	The **total of 12 teams participated** in competition.
(2)	REF	Raymond zilinskas stated that in the worst case the bacteria would be defrosted from minus 70 degrees and it would be a real mess to clean up afterward because **it would not be known for certain whether all the bacteria was dead**.
	XENT	Raymond Zilinskas stated that the bacterium was defrost in the worst case and that afterward cleaning up was a real mess because **there is certainly no known cause of death for all the bacteriums**.
	F_{BERT}	Raymond Zilinskas stated that the bacterium was defrosting in the worst case and the afterward cleaning up was a real mess because **the bacterium was certainly not known to die of all the bacteriums**.
	F_{DISC}	Raymond Zilinskas stated that the bacterium was defrost in the worst case and the afterward cleaning up was a real mess because **the bacterium was certainly not known to have all died**.

Table 2: Model-generated examples from three of the five explored systems.

learning rate of $1 \cdot 10^{-4}$ and a batch size of 16. Following Ribeiro et al. (2021), we use a linearly decreasing schedule for the learning rate and no warm-up. Since Ribeiro et al. (2021) do not release their training methodology, we train until validation BLEU does not increase for three epochs—an approach found in previous work fine-tuning T5 for AMR-to-text generation (Hoyle et al., 2021). We use SciPy[2] and the Python Optimal Transport library[3] to solve Eqs. 1 and 3.

5.2 Results

Table 1 shows that F_{DISC} achieves the highest scores on all metrics, surpassing F_{CONT} as well. It scores higher than XENT by 1.28 BLEU and 0.71 BLEURT points. Although BL-R was specially trained to optimize BLEU, F_{DISC} still outperforms it by over half a point on that metric.

There is a clear hierarchy among the approaches based on F1 score, with F_{DISC} above F_{CONT}, followed by F_{BERT} at the bottom. This dynamic suggests that the optimized alignments may provide higher quality reward signals during training.

We note that although F_{CONT} performed comparably to BL-R, it could exploit tensor operations and was far faster to compute than BLEU. On the other hand, F_{BERT} achieved significantly lower scores than BL-R. As noted in §2, perhaps the clipped precision counts in BLEU gave BL-R an advantage over the greedy nature of F_{BERT}.

5.3 Analysis

Training stability As shown in Fig. 1, F_{DISC} continues to improve on validation BLEU long after XENT overfits at epoch 18. This runs counter to the expectation of unstable RL-based training. It is also interesting that while F_{CONT} validation performance looks fairly low relative to BL-R, it achieves similar scores at test time. This may be due to irrelevant differences between the validation and test sets, however.

Manual inspection Table 2 lists a few examples of model outputs for detailed analysis. In example (1), both XENT and F_{BERT} make the error of predicting “part” instead of “participating”. Only F_{DISC} approaches the meaning of the reference. This may be a side-effect of weighting lexical over semantic similarity in the former two systems. In (2), F_{BERT} repeats the word “bacterium”, while XENT takes an anthropomorphic view of the bacterium. The repetition may be a result of F_{BERT} rewarding multiple instances of the same token by mistake during greedy alignment.

6 Conclusion

This paper proposes new F1 score metrics based on optimized rather than greedy alignments between predicted and reference tokens. Instead of letting hypotheses align to reference tokens without regard to their frequencies (and vice versa), we extract alignments as a constrained optimization problem. In the discrete case, we treat alignment as a matching problem between hypothesis and reference tokens. In the continuous case, we find alignments that minimize earth mover’s distance between the two token embedding distributions.

We apply new metrics as rewards during RL-based training for AMR-to-text generation, with F_{DISC} outperforming both a cross-entropy baseline and one optimizing BLEU rewards. Despite being computed on a downstream model’s token embeddings, the metrics still provide informative rewards during training without signs of overfitting.

[2] https://scipy.org
[3] https://pythonot.github.io

Acknowledgments Research supported by NSF awards IIS-1813823 and CCF-1934962.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

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

Elizabeth Clark, Asli Celikyilmaz, and Noah A. Smith. 2019. Sentence mover's similarity: Automatic evaluation for multi-sentence texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760.

David F. Crouse. 2016. On implementing 2D rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in Neural Information Processing Systems*, 26:2292–2300.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 355–364.

Alexander Miserlis Hoyle, Ana Marasović, and Noah A Smith. 2021. Promoting graph awareness in linearized graph-to-text generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 944–956.

Inigo Jauregi Unanue, Jacob Parnell, and Massimo Piccardi. 2021. BERTTune: Fine-tuning neural machine translation with BERTScore. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 915–924.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR-15)*.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 957–966.

Siyao Li, Deren Lei, Pengda Qin, and William Yang Wang. 2019. Deep reinforcement learning with distributional semantic rewards for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6038–6044.

Chin-Yew Lin and Eduard Hovy. 2002. Manual and automatic evaluation of summaries. In *Proceedings of the ACL-02 Workshop on Automatic Summarization*, pages 45–51.

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

Gabriel Peyré and Marco Cuturi. 2019. Computational optimal transport: With applications to data science. *Foundations and Trends in Machine Learning*, 11(5-6):355–607.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395.

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

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations*.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892.

Cédric Villani. 2009. *Optimal Transport: Old and New*. Springer, Berlin.

John Wieting, Taylor Berg-Kirkpatrick, Kevin Gimpel, and Graham Neubig. 2019. Beyond BLEU: Training neural machine translation with semantic similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4344–4355.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256.

Sho Yokoi, Ryo Takahashi, Reina Akama, Jun Suzuki, and Kentaro Inui. 2020. Word rotator's distance. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2944–2960.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578.

An Analysis of Negation in Natural Language Understanding Corpora

Md Mosharaf Hossain,[θ] **Dhivya Chinnappa,**[υ] and **Eduardo Blanco**[ϒ]

[θ]Department of Computer Science and Engineering, University of North Texas

[υ]Thomson Reuters

[ϒ]School of Computing and Augmented Intelligence, Arizona State University

mdmosharafhossain@my.unt.edu dhivya.infant@gmail.com eduardo.blanco@asu.edu

Abstract

This paper analyzes negation in eight popular corpora spanning six natural language understanding tasks. We show that these corpora have few negations compared to general-purpose English, and that the few negations in them are often unimportant. Indeed, one can often ignore negations and still make the right predictions. Additionally, experimental results show that state-of-the-art transformers trained with these corpora obtain substantially worse results with instances that contain negation, especially if the negations are important. We conclude that new corpora accounting for negation are needed to solve natural language understanding tasks when negation is present.

1 Introduction

Natural language understanding (NLU) is an umbrella term used to refer to any task that requires text understanding. For example, question answering (Rajpurkar et al., 2016), information extraction (Stanovsky et al., 2018), coreference resolution (Wu et al., 2020), and machine reading (Yang et al., 2019), among many others, are tasks that fall under natural language understanding. The threshold for claiming that a system understands natural language is ever-moving. New corpora are often justified by pointing out that state-of-the-art models do not obtain good results. After years of steady improvements, more powerful models eventually obtain so-called human performance, and at that point new, more challenging corpora are created.

Many corpora for natural language understanding tasks contain language generated by annotators rather than retrieved from texts written independently of the corpus creation process. These corpora are certainly useful and have facilitated tremendous progress. Annotator-generated examples, however, carry the risk of evaluating systems with synthetic language that is not representative of language in the wild. For example, annotators are likely to use negation when asked to write a text that contradicts something despite contradictions in the wild need not have a negation (Gururangan et al., 2018). Recently, Kwiatkowski et al. (2019) present a large corpus for question answering that consists of natural questions (i.e., asked by somebody with a real information need) in order to encourage research in a more realistic scenario. This contrasts with previous corpora, where the questions were written by annotators after being told the answer (Rajpurkar et al., 2016).

In this paper, we explore the role of negation in eight corpora for six popular natural language understanding tasks. Our goal is to check whether negation plays the role it deserves in these tasks. To our surprise, we conclude that negation is virtually ignored by answering the following questions:[1]

1. Do NLU corpora contain as many negations as general-purpose texts? (they don't);
2. Do the (few) negations in NLU corpora play a role in solving the tasks? (they don't); and
3. Do state-of-the-art transformers trained with NLU corpora face challenges with instances that contain negation? (they do, especially if the negation is important).

2 Background and Related Work

We work with the eight corpora covering six tasks summarized below and exemplified in Table 2.

We select two corpora for question answering: CommonsenseQA (Talmor et al., 2019) and COPA (Roemmele et al., 2011). CommonsenseQA consists of multi-choice questions (5 candidate answers) that require some degree of commonsense. COPA presents a premise (e.g., *The man broke his toe*) and a question (e.g., *What was the cause of this?*) and the system must choose between two plausible alternatives (e.g. *He got a hole in his sock* or *He dropped a hammer on his foot*).

[1] Code and data available at https://github.com/mosharafhossain/negation-and-nlu.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 716 - 723
May 22-27, 2022 ©2022 Association for Computational Linguistics

For textual similarity and paraphrasing, we select QQP[2] and STS-B (Cer et al., 2017). QQP consists of pairs of questions and the task is to determine whether they are paraphrases. STS-B consists of pairs of texts and the task is to determine how semantically similar they are with a score from 0 to 5.

We select one corpus for the remaining tasks. For inference, we work with QNLI (Rajpurkar et al., 2016), which consists in determining whether a text is a valid answer to a question. We use WiC (Pilehvar and Camacho-Collados, 2019) for word sense disambiguation. WiC consists in determining whether two instances of the same word (in two sentences; italicized in Table 2) are used with the same meaning. For coreference resolution, we choose WSC (Levesque et al., 2012), which consists in determining whether a pronoun and a noun phrase are co-referential (italicized in Table 2). Finally, we work with SST-2 (Socher et al., 2013) for sentiment analysis. The task consists in determining whether a sentence from a collection of movie reviews has positive or negative sentiment.

For convenience, we work with the formatted versions of these corpora in the GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019) benchmarks. The only exception is CommonsenseQA, which is not part of these benchmarks.

Related Work Previous work has shown that SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018) have annotation artifacts (e.g., negation is a strong indicator of *contradictions*) (Gururangan et al., 2018). The literature has also shown that simple adversarial attacks including negation cues are very effective (Naik et al., 2018; Wallace et al., 2019). Kovatchev et al. (2019) analyze 11 paraphrasing systems and show that they obtain substantially worse results when negation is present.

More recently, Ribeiro et al. (2020) show that negation is one of the linguistic phenomena commercial sentiment analysis struggle with. Several previous works have investigated the (lack of) ability of transformers to make inferences when negation is present. For example, Ettinger (2020) conclude that BERT is unable to complete sentences when negation is present. BERT also faces challenges solving the task of natural language inference (i.e., identifying entailments and contradictions) with monotonicity and negation (Geiger et al., 2020; Yanaka et al., 2019). Warstadt et al. (2019) show the limitations of BERT making acceptability judgments with sentences that contain negative polarity items. Most related to out work, Hossain et al. (2020) analyze the role of negation in three natural language inference corpora: RTE (Dagan et al., 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), SNLI and MNLI. In this paper, we present a similar analysis, but we move beyond natural language inference and work with eight corpora spanning six natural language understanding tasks.

	#sents.	% w/ neg.
Question Answering		
CommonsenseQA	12,102	14.5
COPA	1,000	0.8
Similarity and Paraphrasing		
QQP	1,590,482	8.1
STS-B	17,256	7.1
Inference		
QNLI	231,338	8.7
Word Sense Disambiguation		
WiC	14,932	8.2
Coreference Resolution		
WSC	804	26.2
Sentiment Analysis		
SST-2	70,042	16.0
General-purpose English		
all sentences	8,300,000	22.6–29.9
only questions	456,214	15.8–20.2

Table 1: Number of sentences and percentage of sentences containing negation in natural language understanding corpora. All but WSC contain substantially fewer negations than general-purpose English texts.

3 Research Questions and Analysis

Q1: Do natural language understanding corpora contain as many negations as general-purpose English texts? In order to automatically identify negation cues, we train a negation cue detector with the largest corpus available, ConanDoyle-neg (Morante and Daelemans, 2012). The cue detector is based on the RoBERTa pretrained language model (Liu et al., 2019); we provide details about the architecture and training process in Appendix A. Our cue detector obtains the best results to date: F1: 93.79 vs. 92.94 (Khandelwal and Sawant, 2020). ConanDoyle-neg (and thus our cue detector) identifies common negation cues such as *no*, *not*, *n't* and *never*, affixal negation cues such as *impossible* and *careless*, and lexical negations such as *deny* and *avoid*.

[2] https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs

	Example		Important?
CmmnsQA	[. . .] he (John) never saw the lady before. They were what? A) pay debts, B) slender, C) unacquainted, D) free flowing, E) sparse	C	✓
	When you travel you should what in case of unexpected costs? A) go somewhere, B) energy, C) spend frivilously, D) fly in airplane, E) have money	E	✗
QQP	What are some not-so-boring baby shower games ? What are some baby shower games that are actually fun?	yes	✓
	Who was philosophical guru of Shivaji Maharaj? What are the unknown facts of shivaji maharaj?	no	✗
STS-B	Colin Powell, the Secretary of State, said contacts with Iran would not stop. Secretary of State Colin Powell said yesterday that contacts with Iran would continue.	4.3	✓
	Well for one a being could have a non-physical existance and yet not even be in your mind. The difference is huge, as not all non-physical things exist in minds.	3.4	✗
QNLI	Who did BSkyB team up with as it was not part of consortium? While BSkyB had been excluded from being a part of the [. . .], BSkyB was able to join ITV Digital's free-to-air replacement, Freeview, in which it holds an equal stake [. . .]	yes	✓
	In what year did Lavoisier publish his work on combustion? In one experiment, Lavoisier observed that there was no overall increase in weight when tin and air were heated in a closed container.	no	✗
SST-2	It's not the ultimate depression-era gangster movie.	neg.	✓
	Whaley's determination to immerse you in sheer, unrelenting wretchedness is exhausting.	neg.	✗
WiC	The *intention* of this legislation is to boost the economy. Good *intentions* are not enough.	same	✗
WSC	*Sam and Amy* are passionately in love, but Amy's parents are unhappy about it, because *they* are only fifteen.	yes	✗

Table 2: Examples containing negation (underlined) from the validation datasets of the natural language understanding corpora we work with. The third column presents the expected answer for the example (a choice, judgment, or score depending on the task). The last column indicates whether the negation is important.

Table 1 presents the percentage of sentences that contain negation in (a) the eight corpora we work with and (b) general-purpose English. We take the latter percentage (all sentences) from Hossain et al. (2020), who run a negation cue detector in online reviews, conversations, and books. Additionally, we also present the percentages in questions. Negation is much less common in all natural language understanding corpora but WSC (0.8%–16%) than in general-purpose English (22.6%–29.9%). Note that negation is also underrepresented in corpora that primarily contain questions (general-purpose: 15.8%–20.2%; COPA: 0.8%, QQP: 8.1%).

Q2: Do the (few) negations in natural language understanding corpora play a role in solving the tasks? After showing that negation in underrepresented in natural language understanding corpora, we explore whether the few negations they contain are important. Given an instance from any of the corpora, we consider a negation *important* if removing it changes the ground truth. In other words, a negation is *unimportant* if one can ignore it and still solve the task at hand. Table 2 presents examples of important and unimportant negations.

We manually examine the negations in all instances containing negation from the validation split of each corpus except QQP, for which we examine 1,000 (out of 5,196). Note that COPA does not have any negations in the validation split, and many corpora have few instances containing negation (CommonsenseQA: 184, STS-B: 225, QNLI: 852, WiC: 99, WSC: 52, and SST-2: 263). We choose to work with the validation set because we want to compare results when negation is and is not important (Q3), and the ground truth for the test splits of some corpora are not publicly available.

We observe that (a) all negations in WiC and WSC are unimportant, and (b) the percentages of unimportant negations in CommonsenseQA, SST-2, QQP, STS-B, and QNLI are substantial: 45.1%, 63%, 97.4%, 95.6%, and 97.7%, respectively. These percentages indicate that one can safely ignore (almost) all negations and still solve the benchmarks. Despite the fact that negations are

		Example		Important?
CommonsenseQA	Syntactic	Where would a person live if they wanted no neighbors? A) housing estate, B) neighborhood, C) mars, D) woods, E) suburbs	D	✓
		The teacher doesn't tolerate noise during a test in their what? A) movie theatre, B) bowling alley, C) factory, D) store, E) classroom	E	✗
	Morpho.	What might result in an unsuccessful suicide attempt? A) die, B) interruption, C) bleed, D) hatred, E) dying	B	✓
		How are the conditions for someone who is living in a homeless shelter? A) sometimes bad, B) happy, C) respiration, D) growing older, E) death	A	✗
STS-2	Syntactic	Despite the evocative aesthetics evincing the hollow state of modern love life, the film never percolates beyond a monotonous whine.	neg.	✓
		Even if you don't think (kissinger's) any more guilty of criminal activity than most contemporary statesmen, he'd sure make a courtroom trial great fun to watch.	pos.	✗
	Morpho.	Makes for a pretty unpleasant viewing experience.	neg.	✓
		For anyone unfamiliar with pentacostal practices in general and theatrical phenomenon of hell houses in particular, it's an eye-opener .	pos.	✗

Table 3: Examples containing syntactic and morphological negation (underlined) from the validation datasets of CommonsenseQA and SST-2.

	CmmnsnsQA	COPA	QQP	STS-B	QNLI	WiC	WSC	SST-2
validation w/o neg	0.60	0.73	0.90	0.92 / 0.91	0.93	0.67	0.63	0.94
validation w/ neg	0.53	n/a	0.91	0.85 / 0.84	0.91	0.64	0.59	0.93
important (sample from Q2)	0.47	n/a	0.73	0.57 / 0.62	0.67	n/a	n/a	0.86
unimportant (sample from Q2)	0.62	n/a	0.92	0.85 / 0.84	0.92	0.64	0.59	0.95

Table 4: Results obtained with RoBERTa evaluating against (a) all instances with and without negation, and (b) the sample of instances with negation we analyze in detail (important and unimportant). Since the datasets are unbalanced, we report macro F1-score for all tasks except STS-B, for which we report Pearson and Spearman correlations. Results are slightly lower with negation, and substantially lower with *important* negations.

not important in WSC and WiC, they do affect the experimental results (details in Q3).

We also analyze the role of two major types of negation: syntactic (*not*, *no*, *never*, etc.) and morphological (i.e., affixes such as *un-*, *im-*, and *-less*). To this end, we work with CommonsenseQA and SST-2, which have lower percentages of unimportant negations (45.1% and 63%) than the other corpora we use (97.4%–100%). Table 3 provides examples of these two negation types. Perhaps unsurprisingly, syntactic negations are much more common than morphological negations (CommonsenseQA: 88.6% vs 11.4%, SST-2: 71.9% vs 28.1%). More importantly, syntactic negations are more often important in SST-2 (42.3% vs 23%), but both syntactic and morphological negation are roughly equaly important in CommonsenseQA (55.2% vs 52.4%).

Q3: Do state-of-the-art transformers trained with NLU corpora face challenges with instances that contain negation? We conduct experiments with RoBERTa (Liu et al., 2019). More specifically, we use the implementation by Phang et al. (2020) and train a model with the training split of each corpus. We refer the readers to the Appendix B for the details about these models and hyperparameters. We chose RoBERTa over other transformers because 4 out of the 10 best submissions to the SuperGLUE benchmark use it.[3]

Table 4 presents the results evaluating the models with the corresponding validation splits. RoBERTa obtains slightly worse results with the validation instances that have negation in all corpora; the only exception is QQP (F1: 0.90 vs. 0.91). These results lead to the conclusion that negation *may* only pose a small challenge to state-of-the-art transformers.

The results obtained evaluating with the important and unimportant negations from the samples analyzed in Question 2, however, provide a different picture. Indeed, we observe substantial drops in results in all tasks that have both kinds of negations. More specifically, we obtain 27% lower results

[3] https://super.gluebenchmark.com/leaderboard

with instances containing important negations in QNLI (F1: 0.92 vs. 0.67), 33%/26% lower in STS-B, 24% lower in CommonsenseQA, 21% lower in QQP, and 9% lower in SST. Further, even though all negations are unimportant in WiC and WSC, we observe a drop in performance for the instances with negation compared to the instances without negation (WiC: 0.64 vs 0.67 and WSC: 0.59 vs 0.63). We conclude that transformers trained with existing NLU corpora face challenges with instances that contain negation. These results raise two important questions for future research: Is negation an inherently challenging phenomenon for RoBERTa? How many instances with negation are required to solve a natural language understanding task?

4 Conclusions

We have analyzed the role of negation in eight natural language understanding corpora covering six tasks. Our analyses show that (a) all but WSC contain almost no negations or around 31%–54% of the negations found in general-purpose texts, (b) the few negations in these corpora are usually unimportant, and (c) RoBERTa obtains substantially worse results when negation is important.

Our analyses also provide some evidence that creating models to properly deal with negation may require both new corpora and more powerful models. The need for new corpora stems from the answers to Questions 1 and 2. The justification for powerful models is more subtle. We point out that the percentage of unimportant negations (Section 3) is only a weak indicator of the drop in results with important negations (Table 4). For example, we observe a 24% and 21% drop in results with important negations from CommonsenseQA and QQP despite 45% and 97% of negations are unimportant.

Negation reverses truth values thus solutions to any natural language understanding task should be robust when negation is present and important. To this end, our future work includes two lines of research. First, we plan to create benchmarks for the six tasks consisting of instances containing negation (50/50 split important/unimportant). Second, we plan to conduct probing experiments to investigate whether (and where) pretrained transformers capture the meaning of negation. Doing so may help us discover potential solutions to understand negation and make inferences.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1845757. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF. The Titan Xp used for this research was donated by the NVIDIA Corporation. Computational resources were also provided by the UNT office of High-Performance Computing. Further, we utilized computational resources from the Chameleon platform (Keahey et al., 2020). We also thank the reviewers for insightful comments.

References

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge. In *Proceedings of the second PASCAL challenges workshop on recognising textual entailment*, volume 6, pages 6–4. Venice.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge.

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

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, MLCW'05, pages 177–190, Berlin, Heidelberg. Springer-Verlag.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Atticus Geiger, Kyle Richardson, and Christopher Potts. 2020. Neural natural language inference models partially embed theories of lexical entailment and negation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 163–173, Online. Association for Computational Linguistics.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. 2020. An analysis of natural language inference benchmarks through the lens of negation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9106–9118, Online. Association for Computational Linguistics.

Kate Keahey, Jason Anderson, Zhuo Zhen, Pierre Riteau, Paul Ruth, Dan Stanzione, Mert Cevik, Jacob Colleran, Haryadi S. Gunawi, Cody Hammock, Joe Mambretti, Alexander Barnes, François Halbach, Alex Rocha, and Joe Stubbs. 2020. Lessons learned from the chameleon testbed. In *Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC '20)*. USENIX Association.

Aditya Khandelwal and Suraj Sawant. 2020. NegBERT: A transfer learning approach for negation detection and scope resolution. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5739–5748, Marseille, France. European Language Resources Association.

Venelin Kovatchev, M. Antonia Marti, Maria Salamo, and Javier Beltran. 2019. A qualitative evaluation framework for paraphrase identification. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 568–577, Varna, Bulgaria. INCOMA Ltd.

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

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'12, page 552–561. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Roser Morante and Walter Daelemans. 2012. ConanDoyle-neg: Annotation of negation cues and their scope in conan doyle stories. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 1563–1568, Istanbul, Turkey. European Language Resources Association (ELRA).

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jason Phang, Phil Yeres, Jesse Swanson, Haokun Liu, Ian F. Tenney, Phu Mon Htut, Clara Vania, Alex Wang, and Samuel R. Bowman. 2020. `jiant` 2.0: A software toolkit for research on general-purpose text understanding models. `http://jiant.info/`.

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

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

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, pages 90–95.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 885–895, New Orleans, Louisiana. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32*, pages 3261–3275. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, Sheng-Fu Wang, Jason Phang, Anhad Mohananey, Phu Mon Htut, Paloma Jeretic, and Samuel R. Bowman. 2019. Investigating BERT's knowledge of language: Five analysis methods with NPIs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2877–2887, Hong Kong, China. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6953–6963, Online. Association for Computational Linguistics.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, Kentaro Inui, Satoshi Sekine, Lasha Abzianidze, and Johan Bos. 2019. HELP: A dataset for identifying shortcomings of neural models in monotonicity reasoning. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 250–255, Minneapolis, Minnesota. Association for Computational Linguistics.

An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2346–2357, Florence, Italy. Association for Computational Linguistics.

A Negation Cue Detection

We develop a negation cue detector (Section 3 in the paper) by utilizing the RoBERTa (base architecture; 12 layers) pre-trained model (Liu et al., 2019). We fine-tune the system on ConanDoyle-neg (Morante and Daelemans, 2012) corpus. While fine-training, the negation cues are marked with BIO (B: Beginning of cue, I: Inside of cue, O: Outside of cue) tagging scheme. The contextualized representations from the last layer of RoBERTa are passed to a fully connected (FC) layer. Finally, a conditional random field (CRF) layer produces the output sequence for the labels.

Our model yields the following results on the test set: 93.26 Precision, 94.32 Recall, and 93.79 F1. The neural model takes about two hours on average to train on a single GPU of NVIDIA Tesla K80. A list of the tuned hyperparameters that the model requires to achieve the above results is provided in Table 5. The code is available at `https://github.com/mosharafhossain/negation-and-nlu`.

Hyperparameter	
Max Epochs	50
Batch Size	10
Learning Rate (RoBERTa)	1e-5
Learning Rate (FC, CRF)	1e-3
Weight Decay (RoBERTa)	0.00001
Weight Decay (FC)	0.001
Grad Clipping	5.0
Warmup Epochs	5
Patience	15
Dropout	0.5

Table 5: Hyperparameters used to fine-tune the cue detector with ConanDoyle-neg (Morante and Daelemans, 2012) corpus. FC and CRF refers to fully connected and conditional random field layers, respectively.

	Hp-1	Hp-2	Hp-3
CmmnsnsQA	10	16	1e-5
COPA	50	16	1e-5
QQP	3	16	1e-5
STS-B	10	16	1e-5
QNLI	3	8	1e-5
WiC	10	16	1e-5
WSC	200	16	1e-6
SST-2	3	16	1e-5

Table 6: Hyperparameters used to fine-tune RoBERTa individually for each corpus. Hp-1, Hp-2, and Hp-3 refer to the number of epochs, batch size, and learning rate used in the training procedure. We use default settings for the other hyperparameters when we use the implementation by Phang et al. (2020).

B Hyperparameters to Fine-tune the System for Each of the NLU Tasks

We use an implementation by Phang et al. (2020) and fine-tune RoBERTa (base architecture; 12 layers) (Liu et al., 2019) model separately for each of the eight corpora. We use the default settings of the hyperparameters, except for a few, when fine-tuning the model on each benchmark. Table 6 shows tuned hyperparameters for each benchmark.

Primum Non Nocere: Before working with Indigenous data, the ACL must confront ongoing colonialism

Lane Schwartz

Department of Linguistics
University of Illinois
Urbana, Illinois
lanes@illinois.edu

Institute of Northern Engineering
University of Alaska
Fairbanks, Alaska
loschwartz@alaska.edu

Abstract

In this paper, we challenge the ACL community to reckon with historical and ongoing colonialism by adopting a set of ethical obligations and best practices drawn from the Indigenous studies literature. While the vast majority of NLP research focuses on a very small number of very high resource languages (English, Chinese, etc), some work has begun to engage with Indigenous languages. No research involving Indigenous language data can be considered ethical without first acknowledging that Indigenous languages are not merely very low resource languages. The toxic legacy of colonialism permeates every aspect of interaction between Indigenous communities and outside researchers. Ethical research must actively challenge this colonial legacy by actively acknowledging and opposing its continuing presence, and by explicitly acknowledging and centering Indigenous community goals and Indigenous ways of knowing. To this end, we propose that the ACL draft and adopt an ethical framework for NLP researchers and computational linguists wishing to engage in research involving Indigenous languages.

1 Introduction

Beginning with our community's first academic conference in 1952 (see Reifler, 1954) and continuing with the establishment of the Association for Computational Linguistics (ACL)[1] in 1962 (MT Journal, 1962), the members of our research community have examined a huge range of topics, ranging from linguistic and computational linguistic models and theories to engineering-focused problems in natural language processing.[2]

While great progress has been made in recent years across many NLP tasks, the overwhelming majority of NLP and CL research focuses on a very small number of languages. Over the 70 years from 1952 to 2022, the vast majority of CL and NLP research has focused on a small number of widely-spoken languages, nearly all of which represent politically- and economically-dominant nation-states and the languages of those nation-states' historical and current adversaries: English, the Germanic and Romance languages of western Europe, Russian and the Slavic languages of eastern Europe, Hebrew, Arabic, Chinese, Japanese, and Korean. Bender (2009) surveyed papers from ACL 2008 and found that English dominated (63% of papers), with 20 other languages distributed along a Zipfian tail (Chinese and German shared the number 2 slot at just under 4% of papers each); across all ACL 2008 long papers, only three languages (Hindi, Turkish, and Wambaya) were represented outside of the language families listed previously. This lack of diversity directly impacts both the quality and ethical status of our research, as nearly every successful NLP technique in widespread current use was designed around the linguistic characteristics of English.[3]

A special theme designed to address this shortcoming has been selected for the 60th Annual Meeting of the ACL in 2022: *"Language Diversity: from Low Resource to Endangered Languages."* This theme is to be commended as a step towards a more linguistically diverse research agenda. Yet as we expand our research to a broader and more inclusive set of languages, we must take great care to do so ethically. The endangered Indigenous languages of the world are not merely very low resource languages. The toxic legacy of colonial-

[1] Originally founded as the Association for Machine Translation and Computational Linguistics, the current name stems from 1968 after the publication of the 1966 ALPAC report.

[2] See Linguistic Issues in Language Technology (2011) and Eisner (2016) for excellent discussions on the distinction between computational linguistics (CL) and natural language processing (NLP).

[3] A small minority of successful NLP techniques were designed taking into account the characteristics of a few other languages, nearly all from the Indo-European and Sino-Tibetan language families.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 724 - 731
May 22-27, 2022

ism permeates every aspect of interaction between Indigenous communities and outside researchers (Smith, 2012). Ethical research must actively challenge this colonial legacy by actively acknowledging and opposing its continuing presence, and by explicitly acknowledging and centering Indigenous community goals and Indigenous ways of knowing.

To this end, we propose an ethical framework for NLP researchers and computational linguists wishing to engage in research involving Indigenous languages. We begin in §2 by examining the abstracts of papers published in the proceedings of the top-tier conferences (ACL, NAACL, EMNLP, EACL, AACL) and journals (Computational Linguistics, TACL) of the Association for Computational Linguistics from the past several years (hereafter referred to as *ACL papers/abstracts), replicating the results of Bender (2009), confirming that recent *ACL papers still lack significant language diversity. In §3 we address research practices and ongoing colonialism in Indigenous communities. Finally, we examine decolonial practices appropriate for a draft framework of ethical obligations (§4) for the ACL research community.

2 Recent *ACL papers lack significant language diversity

We begin by examining the abstracts of *ACL papers from the past several years to confirm the results of Bender (2009), namely that recent *ACL papers still lack significant language diversity. We collect a corpus of 9602 recent *ACL abstracts from the ACL Anthology;[4] more than 80% fail to mention any language (see Table 1). Essentially all such papers that fail the #BenderRule assume English as the language of study (Bender, 2019). Vanishingly few abstracts mention any Indigenous language. While 66 abstracts mention Arabic, fewer than 20 abstracts mention any other African language. Only 11 abstracts mention any Indigenous language of North America. Only 2 abstracts mention an Indigenous language of Australia. Only 1 abstract mentioned an Indigenous language of Te Riu-a-Māui. No abstracts mentioned any Indigenous language of South America.

Table 1 shows a Zipfian distribution predominated by four language families: Indo-European (dominated by English), Sino-Tibetan (dominated by Mandarin Chinese), Japonic (essentially all Japanese), and Afro-Asiatic (dominated by Arabic and Hebrew). Indo-European languages are assumed (English) or explicitly mentioned in 97% of abstracts. The next three most mentioned language families account for another 1% of abstracts.[5] Combined, only 165 out of 9602 abstracts (1.7%) mention any language from any other language family.

83.26%	7995	Implictly assume English
13.70%	1315	Indo-European (incl. English)
4.50%	432	Sino-Tibetan
1.12%	108	Japonic
0.85%	82	Afro-Asiatic
0.41%	39	Turkic
0.26%	25	Koreanic
0.25%	24	Austroasiatic
0.24%	23	Dravidian
0.22%	21	Uralic
0.21%	20	Austronesian
0.09%	9	Basque
0.09%	9	Atlantic-Congo
0.07%	7	Na-Dene
0.05%	5	Kra-Dai
0.02%	2	Arnhem
0.02%	2	Iroquoian
0.02%	2	Inuit-Yupik-Unangan
0.01%	1	Sumerian

Table 1: Of 9602 *ACL abstracts (2013–Nov. 2021),[4] percentage and number of abstracts that explicitly mention at least one language from the language family.

These findings are also consistent with those of Joshi et al. (2020), who scrape and examine a corpus of approximately 44,000 papers, including both *ACL papers and papers from LREC, COLING, and ACL-affiliated workshops. Joshi et al. present a 6-point taxonomy for classifying languages according to the quantity of labelled and unlabelled corpora and models available for each language, and find that *ACL papers are low in terms of language diversity and are dominated by the highest-resource languages. Unfortunately, we were unable to apply our language family-level analysis on their dataset, as it was not publicly available for down-

[4] Since 2013, the ACL Anthology has included abstracts for TACL papers. Since 2017, the ACL Anthology has included abstracts for papers published at ACL, EACL, AACL, NAACL, EMNLP, and the Comptuational Linguistics journal. See Appendix A for details.

[5] Note that this is less than the percentages for these three language families listed in Table 1. This is because some abstracts mention multiple languages. This additional 1% represents abstracts that mentioned a language from the Sino-Tibetan, Japonic, or Afro-Asiatic language families and did not also mention an Indo-European language such as English.

load. While Joshi et al. (2020) find that language diversity is somewhat higher at LREC and ACL-affiliated workshops, the larger issue of language homogeneity in top-tier *ACL venues is extremely problematic. In a research community that calls itself the *Association for Computational Linguistics*, it is completely unacceptable that fewer than 20% of top-tier *ACL abstracts mention the name of any language (see Table 1), and those that do are dominated by one language (English) and its language family (Indo-European).

3 Research and Ongoing Colonialism in Indigenous Communities

The linguistic homogeneity in *ACL papers can be viewed as a symptom of a much larger problem, namely that our research paradigms are deeply rooted in a Western scientific tradition that is inextricably intertwined with colonialism. Smith (2012, p.50) notes that in this tradition, there are implicit and explicit rules of framing and practice that express power. In *ACL research, the act of not explicitly stating any language, of assuming English as the default, is one such practice.

Research scientists rarely consider the philosophy of science (Popper, 1959) on which our research is predicated; as Wilson (2001) notes, this is defined by an ontology, epistimology, methodologies, and axiology that are seldom acknowledged. In our field, these often surface as unacknowledged positivist (Comte, 1853) assumptions that science is value-neutral and empirical observations and logical reasoning fully and completely define the nature of science and reality (Egan, 1997). The first step in enacting decolonial ethical practices is acknowledging that we hold these assumptions and recognizing that there are other Indigenous philosophies of science that are equally valid and are rooted in fundamentally distinct worldviews that center relationality (see Wilson, 2008). By failing to acknowledge and critically examine the philosophical foundations of our science, we implicitly and unconsciously elevate our ideas of research and language work above those of Indigenous communities (Leonard, 2017).

Given the distinct value systems and distinct views of reality of outside research scientists and Indigenous communities, it is not surprising that even good-faith efforts of well-meaning outside researchers are often viewed by Indigenous communities as irrelevant at best and exploitative at worst.[6] Outside perceptions of Indigenous peoples are inextricably linked to corresponding histories of colonization, and are typically accompanied by (usually outdated and incorrect) assumptions about the "proper" roles of Indigeneous peoples today that correspond with neither reality nor Indigenous people's views of themselves (Deloria, 2004; Leonard, 2011). When a linguist (or a computer scientist) begins the process of interacting with an Indigenous community and working with that community's Indigenous language, the starting "lens through which others view [the linguist's] professional activities will at least partly reflect what 'linguist' has come to mean, and that this in some cases will occur regardless of whether [the linguist] personally exhibit a trait that has come to be associated with this named position" (Leonard, 2021).

Endangered Indigenous languages are not merely very low-resource languages. Each Indigenous community represents a sovereign political entity. Each Indigenous language represents a crucial component of the shared cultural heritage of its people. The rate of intergenerational transmission of Indigenous language from parent to child in many Indigeneous communities has declined and is continuing to decline (Norris, 2006), resulting in a deep sense of loss felt by older generations who grew up speaking the Indigenous language as well as by younger generations who do not speak the language who experience a diminished sense of cultural inclusion (Tulloch, 2008). Language is an integral part of culture, and declines in robust Indigenous language usage have been correlated with serious negative health and wellness outcomes (Chandler and Lalonde, 2008; Reid et al., 2019).

At the same time, Indigenous individuals and Indigenous communities have suffered greatly from colonial practices that separated children from communities, actively suppressed Indigenous language and culture, misappropriated land and natural resources, and treated Indigenous people, cultures, and languages as dehumanized data to study (Whitt, 2009; NTRC, 2015; Leonard, 2018; Bull, 2019; Dei, 2019; Guematcha, 2019; Bahnke et al., 2020; Kawerak, 2020). As Smith (2012) notes, "*research*

[6] We note that not all researcher scientists are outsiders from an Indigenous perspective. Indigenous scholars have played and continue to play important roles within numerous fields of scholarship, including linguistics, computational linguistics, natural language processing, and machine learning. (see, for example Lewis, 2020)

is probably one of the dirtiest words in the indigenous world's vocabulary;" it is "implicated in the worst excesses of colonialism" and "told [Indigenous people] things already known, suggested things that would not work, and made careers for people who already had jobs." It is then, hardly surprising that "After generations of exploitation, Indigenous people often respond negatively to the idea that their languages are data ready for the taking" (Bird, 2020).

Indigenous communities are rightly taking up the slogan "Nothing about us without us" (see, for example, Pearson, 2015). Even when we consider the "lived experiences and issues that underlie [the] needs" of Indigenous communities, these community priorities are far too often treated as subordinate to research questions deemed valuable by members of academe (Leonard, 2018; Wilson, 2008; Simonds and Christopher, 2013). Credulous evangelical claims of technology as savior[7,8] only exacerbate these tensions (Irani et al., 2010; Toyama, 2015).

4 Prerequisite Obligations for Ethical Research involving Indigenous Languages and Indigenous Peoples

When CL and NLP researchers begin to work with Indigenous language data without first critically examining the toxic legacy of colonialism and the self-identified priority needs and epistemology of the Indigenous community, the risk of unwittingly perpetuating dehumanizing colonial practices is extremely high. It is therefore critically urgent that the ACL, perhaps through the recently-formed Special Interest Group on Endangered Languages (SIGEL), should go beyond the ACL's 2020 adoption of the ACM Code of Ethics[9] and begin a process of drafting and adopting a formal ethics policy specifically with respect to research involving Indigenous communities, Indigenous languages, and Indigenous data. In so doing, the ACL can provide specific and foundational ethical guidance for our members that goes far beyond the general ethical guidance provided by institutional review boards (only some of which are intimately familiar with the ethical pitfalls particular to work with Indigenous communities).

We should draw upon the recent Linguistics Society of America (2019) ethics statement, the foundational principles of medical ethics (autonomy, nonmaleficence, beneficence, and justice; Beauchamp and Childress, 2001), the recommendations of Bird (2020), and the wisdom of Indigenous scholars such as Deloria, Wilson, Smith, and Leonard.

As a beginning, we have identified four key ethical obligations that should at a minimum be included in such an ethics policy: cognizance, beneficence, accountability, and non-maleficence.

4.1 Obligation of cognizance

The colonial political and racial ideas and behaviors that support and enable colonization and oppression are intentionally invented historical creations (Allen, 2012; Kendi, 2017). Before we engage with Indigenous peoples, let alone work with Indigenous data, we must intentionally make ourselves cognizant of this history. As outside researchers, we stand in a privileged position, and as such have an urgent obligation to educate ourselves about this history and about current practices that perpetuate these systems of oppression in the present day (Kendi, 2019; Smith, 2012).[10]

Before we are capable of ethically engaging with Indigenous data, we must learn the ways in which Indigenous communities approach reality and science, and accept that these are fully formed and fully valid worldviews with which we have an obligation to fully engage. Our research is premised on a particular philosophy of science which is nearly always left unstated. We must make ourselves cognizant of our own ontology, epistemology, methodology, and axiology, and the fact that there are alternative philosophies of science that are equally valid. We must educate ourselves about Indigenous ontologies, epistemologies, methodologies, and axiologies that are centered around relationality (Wilson, 2008).

The *obligation of cognizance* therefore mandates that we as researchers intentionally and thoroughly educate ourselves about colonization of Indigenous communities; about the role that academic researchers have had and continue to play in the

[7] "The number of endangered languages is so large that their comprehensive documentation by the community of documentary linguists will only be possible if supported by NLP technology." (Vetter et al., 2016)

[8] "Languages that miss the opportunity to adopt Language Technologies will be less and less used, while languages that benefit from cross-lingual technologies such as Machine Translation will be more and more used." (ELRA, 2019)

[9] https://www.aclweb.org/portal/content/acl-code-ethics

[10] See, for example, https://www.wired.co.uk/article/maori-language-tech

exploitation of Indigenous communities, Indigenous languages, Indigenous culture, and Indigenous data; and about Indigenous expectations and ways of being centered on relationality that differ from those we typically encounter in our research.

In practical terms, this cognizance and the education requisite in this obligation should typically be provided by a senior researcher (one already very familiar with the relevant issues) whenever a new student or junior researcher first expresses an interest to begin research involving Indigenous data. At an institutional level, the leadership of multilingual NLP shared tasks such as the SIGMORPHON shared tasks should take the lead in educating their respective sub-communities in this regard as such shared tasks consider expansion to include Indigenous language data.

4.2 Obligation of beneficence

Indigenous communities are sovereign political entities with inherent political and human rights. Many of these rights are enumerated in the Declaration on the Rights of Indigenous Peoples (United Nations, 2007). This includes the right of each Indigenous community to protect and develop its culture (Article 11), the right to dignity (Article 15), the right to develop and elect its own decision-making institutions (Article 18), and the right to "maintain, control, protect, and develop [the community's] intellectual property over [its] cultural heritage, traditional knowledge, and traditional cultural expressions" (Article 31).

The *obligation of beneficence* therefore mandates that we as researchers ensure that our work benefits the Indigenous communities with which we work in ways that those communities recognize as beneficial. In practical terms, this means that any outside researcher who wants to work with Indigenous data must seek to engage with the relevant Indigenous communities in order to learn about and to meaningfully support priority areas identified by Indigenous governing bodies and decision-making institutions that fall within our respective scopes of expertise. Put another way, ethical research involving Indigenous data must include concrete deliverables requested by the respective Indigenous community or communities.

4.3 Obligation of accountability

As outside researchers seeking to work with Indigenous data, we have a responsibility to seek out respectful and meaningful relationships with the Indigenous communities whose data we seek to use. We have a responsibility to develop these relationships in ways that are appropriate and meaningful to the Indigenous communities with which we seek to work. We must intentionally acknowledge and accept the rightful authority of Indigenous communities' governing and decision-making bodies over those communities' own respective languages, cultures, and data.

The *obligation of accountability* therefore mandates that we as researchers develop meaningful relations with the sovereign governing bodies of the Indigenous communities with which we seek to engage, and that we be meaningfully accountable to such bodies in our work involving their data. This relationship-building should take place before the research project begins. This relationship between researcher and sovereign Indigenous institutions can be thought of as highly analogous to the relationship between the researcher and governmental granting agencies such as the U.S. National Science Foundation. In practical terms, once this relationship has been built and research has begun, the researcher should regularly report to and agree to be held accountable by Indigenous community's governing and decision-making institutions with respect to the agreed-upon community goals.

4.4 Obligation of non-maleficence

Colonization and colonial practices have inflicted substantial and often genocide-scale harm on Indigenous communities over the past five centuries (Smith, 2017), harm that is ongoing and is often perpetuated by modern research practices.

We must intentionally adopt the ethical prime directive of the medical community, often stated in the Latin aphorism *Primum Non Nocere* "Above all, do no harm" (Smith, 2005). There are many good and laudable reasons why we should choose to engage in research with Indigenous communities, but none of these reasons is powerful enough to justify harm caused by our research.

The *obligation of non-maleficence* therefore mandates that above all else, we do no harm to Indigenous people and Indigenous communities. In practical terms, this means that researchers seeking to engage with Indigenous data critically examine the harmful ramifications of proposed work well before it is conducted. If we can do good through our research without doing harm, that is well, but it is better to not engage than to cause harm.

Acknowledgments

Great thanks are due to the elders and community members of the Alaska Native communities of Gambell and Elim who welcomed me and my family in the 1980s, and from whom I have learned much. To the Yupik and Yup'ik instructors who honored me with a Yupik name and who graciously introduced me to these amazing languages: Igamsiqanaghhalek!

Special thanks are due to the leadership and elected councils and boards of the Native Village of Gambell, Sivuqaq Inc, Gambell Schools, the Bering Strait School District, the Alaska Native Language Archive, and the City of Gambell.

This work was supported by the U.S. National Science Foundation Award #1761680: *NNA: Collaborative Research: Integrating Language Documentation and Computational Tools for Yupik, an Alaska Native Language*.

References

Theodore W. Allen. 2012. *The Invention of the White Race*. Verso Books. Two volumes.

ALPAC. 1966. Language and machines — computers in translation and linguistics. A Report by the Automatic Language Processing Advisory Committee.

Melanie Bahnke, Vivian Korthuis, Amos Philemonoff, and Mellisa Johnson. 2020. Navigating the New Arctic NSF Comment Letter.

Tom L. Beauchamp and James F. Childress. 2001. *Principles of Biomedical Ethics*. Oxford University Press, New York.

Emily Bender. 2019. The #BenderRule: On naming the languages we study and why it matters. *The Gradient*.

Emily M. Bender. 2009. Linguistically naïve != language independent: Why NLP needs linguistic typology. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 26–32, Athens, Greece. Association for Computational Linguistics.

Steven Bird. 2020. Decolonising speech and language technology. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3504–3519, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Julie Bull. 2019. Nothing about us without us: An Inuk reply to exploitive research | Impact Ethics. Memorial University Centre for Bioethics.

Michael J. Chandler and Christopher E. Lalonde. 2008. Cultural continuity as a protective factor against suicide in first nations youth. *Horizons*, 10(1):68–72. Special Issue — Hope or Heartbreak: Aboriginal Youth and Canada's Future.

Auguste Comte. 1853. *The Positive Philosophy of Auguste Comte*. John Chapman, London. Condensed and translated from the original *Cours de Philosophie Positive* (1830–1842) by Harriet Martineau.

George J. Sefa Dei. 2019. Foreword. In *Decolonization and Anti-colonial Praxis*, volume 8 of *Anti-colonial Educational Perspectives for Transformative Change*, pages vii – x. Brill, Leiden, The Netherlands.

Philip J. Deloria. 2004. *Indians in Unexpected Places*. University Press of Kansas, Lawrence.

Kieran Egan. 1997. *The Educated Mind: How Cognitive Tools Shape Our Understanding*. University of Chicago Press.

Jason Eisner. 2016. How is computational linguistics different from natural language processing? Quora.

ELRA. 2019. Lt4all: Language technologies for all – call for participation.

Emmanuel Guematcha. 2019. Genocide against indigenous peoples: The experiences of the truth commissions of Canada and Guatemala. *The International Indigenous Policy Journal*, 10(2).

Lilly Irani, Janet Vertesi, Paul Dourish, Kavita Philip, and Rebecca E. Grinter. 2010. Postcolonial computing: A lens on design and development. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, page 1311–1320, New York, NY, USA. Association for Computing Machinery.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Kawerak. 2020. Knowledge Sovereignty and the Indigenization of Knowledge.

Ibram X. Kendi. 2017. *Stamped From The Beginning: The Definitive History of Racist Ideas in America*. Bold Type Books.

Ibram X. Kendi. 2019. *How to Be an Antiracist*. One World.

Wesley Y. Leonard. 2011. Challenging "extinction" through modern miami language practices. *American Indian Culture and Research Journal*, 35(2):135–160.

Wesley Y. Leonard. 2017. Producing language reclamation by decolonising 'language'. *Language Documentation and Description*, 14:15–36.

Wesley Y. Leonard. 2018. Reflections on (de)colonialism in language documentation. In Bradley McDonnell, Andrea L. Berez-Kroeker, and Gary Holton, editors, *Reflections on Language Documentation 20 Years after Himmelmann 1998*, Special Publication 15, chapter 6, pages 55–65. Language Documentation & Conservation.

Wesley Y. Leonard. 2021. Centering Indigenous ways of knowing in collaborative language work. In Lisa Crowshoe, Inge Genee, Mahaliah Peddle, Joslin Smith, and Conor Snoek, editors, *Sustaining Indigenous Languages: Connecting Communities, Teachers, and Scholars*, pages 21–34. Northern Arizona University.

Jason Edward Lewis, editor. 2020. *Indigenous Protocol and Artificial Intelligence Position Paper*. The Initiative for Indigenous Futures and the Canadian Institute for Advanced Research (CIFAR), Honolulu, Hawaii.

Linguistic Issues in Language Technology. 2011. Interaction of linguistics and computational linguistics. Volume 6.

Linguistics Society of America. 2019. LSA revised ethics statement.

Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui, editors. 2008. *Proceedings of ACL-08: HLT*. Association for Computational Linguistics, Columbus, Ohio.

MT Journal. 1962. Professional society formed. *Mechanical Translation*, 7(1):1.

Mary Jane Norris. 2006. Aboriginal languages in Canada: Trends and perspectives on maintenance and revitalization. In Jerry P. White, Susan Wingert, Dan Beavon, and Paul Maxim, editors, *Aboriginal Policy Research*, volume 3: Moving Forward, Making a Difference, pages 197–226. Thompson Educational Publishing, Toronto.

NTRC. 2015. Honouring the truth, reconciling for the future: Summary of the final report of the Truth and Reconciliation of Canada.

Luke Pearson. 2015. Nothing about us, without us. that's why we need Indigenous-owned media. *The Guardian*.

Karl Popper. 1959. *The Logic of Scientific Discovery*. Hutchinson & Co.

Papaarangi M. Reid, Donna M. Cormack, and Sarah-Jane Paine. 2019. Colonial histories, racism and health — the experience of Māori and Indigenous peoples. *Public Health*, 172:119–124. Special issue on Migration, Ethnicity, Race and Health.

Erwin Reifler. 1954. The first conference on mechanical translation. *Mechanical Translation*, 1(2):23–32.

Vanessa W. Simonds and Suzanne Christopher. 2013. Adapting western research methods to Indigenous ways of knowing. *American Journal of Public Health*, 103(12):2185–2192.

Cedric M. Smith. 2005. Origin and uses of primum non nocere — above all, do no harm! *The Journal of Clinical Pharmacology*, 45(4):371–377.

David Michael Smith. 2017. Counting the dead: Estimating the loss of life in the Indigenous Holocaust, 1492 – present. In *Proceedings of the Twelfth Native American Symposium*.

Linda Tuhiwai Smith. 2012. *Decolonizing Methodologies: Research and Indigenous Peoples*, 2nd edition. Zed Books.

Kentaro Toyama. 2015. *Geek Heresy*. PublicAffairs.

Shelley Tulloch. 2008. Uqausirtinnik annirusunniq — Longing for our language. *Horizons*, 10(1):73–76. Special Issue — Hope or Heartbreak: Aboriginal Youth and Canada's Future.

United Nations. 2007. United Nations declaration on the rights of Indigenous peoples.

Marco Vetter, Markus Müller, Fatima Hamlaoui, Graham Neubig, Satoshi Nakamura, Sebastian Stüker, and Alex Waibel. 2016. Unsupervised phoneme segmentation of previously unseen languages. In *17th Annual Conference of the International Speech Communication Association (InterSpeech 2016)*, San Francisco, California, USA.

Laurelyn Whitt. 2009. *Science, Colonialism, and Indigenous Peoples: The Cultural Politics of Law and Knowledge*. Cambridge University Press.

Shawn Wilson. 2001. What is an Indigenous Research Methodology? *Canadian Journal of Native Education*, 25(2):175–179.

Shawn Wilson. 2008. *Research Is Ceremony: Indigenous Research Methods*. Fernwood Publishing.

A *ACL abstract corpus 2013–Nov. 2021

The *ACL XML files (2013–2021) from the ACL Anthology GitHub repository were downloaded on 6 November 2021.

2013.tacl.xml
2014.tacl.xml
2015.tacl.xml
2016.tacl.xml
2017.acl.xml
2017.cl.xml
2017.eacl.xml
2017.emnlp.xml
2017.tacl.xml
2018.acl.xml
2018.cl.xml
2018.emnlp.xml
2018.naacl.xml
2018.tacl.xml
2019.acl.xml
2019.cl.xml
2019.emnlp.xml
2019.naacl.xml
2019.tacl.xml
2020.aacl.xml
2020.acl.xml
2020.cl.xml
2020.emnlp.xml
2020.tacl.xml
2021.acl.xml
2021.eacl.xml
2021.emnlp.xml
2021.naacl.xml
2021.tacl.xml

The abstracts were extracted from the XML files. From the resulting abstracts all words that begin with an uppercase letter were examined manually to identify all explicitly mentioned language names. All processing steps are described, with specific shell commands used, in the `data` annex that accompanies this paper.

Unsupervised multiple-choice question generation for out-of-domain Q&A fine-tuning

Guillaume Le Berre[1,2], **Christophe Cerisara**[1], **Philippe Langlais**[2], **Guy Lapalme**[2]
[1] University of Lorraine, CNRS, LORIA, France
[2] RALI/DIRO, University of Montreal, Canada
{leberreg, felipe, lapalme}@iro.umontreal.ca, cerisara@loria.fr

Abstract

Pre-trained models have shown very good performances on a number of question answering benchmarks especially when fine-tuned on multiple question answering datasets at once. In this work, we propose an approach for generating a fine-tuning dataset thanks to a rule-based algorithm that generates questions and answers from unannotated sentences. We show that the state-of-the-art model UnifiedQA can greatly benefit from such a system on a multiple-choice benchmark about physics, biology and chemistry it has never been trained on. We further show that improved performances may be obtained by selecting the most challenging distractors (wrong answers), with a dedicated ranker based on a pretrained RoBERTa model.

1 Introduction

In the past years, deep learning models have greatly improved their performances on a large range of question answering tasks, especially using pretrained models such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and T5 (Raffel et al., 2020). More recently, these models have shown even better performances when fine-tuned on multiple question answering datasets at once. Such a model is UnifiedQA (Khashabi et al., 2020), which, starting from a T5 model, is trained on a large number of question answering datasets including multiple choices, yes/no, extractive and abstractive question answering. UnifiedQA is, at the time of writing, state-of-the-art on a large number of question answering datasets including multiple-choice datasets like OpenBookQA (Mihaylov et al., 2018) or ARC (Clark et al., 2018). However, even if UnifiedQA achieves good results on previously unseen datasets, it often fails to achieve optimal performances on these datasets until it is further fine-tuned on dedicated human annotated data. This tendency is increased when the target dataset deals with questions about a very specific domain.

One solution to this problem would be to fine-tune or retrain these models with additionnal human annotated data. However, this is expensive both in time and resources. Instead, a lot of work has been done lately on automatically generating training data for fine-tuning or even training completely unsupervised models for question answering. One commonly used dataset for unsupervised question answering is the extractive dataset SQUAD (Rajpurkar et al., 2016). Lewis et al. (2019) proposed a question generation method for SQUAD using an unsupervised neural based translation method. Fabbri et al. (2020) and Li et al. (2020) further gave improved unsupervised performances on SQUAD and showed that simple rule-based question generation could be as effective as the previously mentioned neural method. These approches are rarely applied to multiple-choice questions answering in part due to the difficulty of selecting distractors. A few research papers however proposed distractor selection methods for multiple-choice questions using either supervised approaches (Sakaguchi et al., 2013; Liang et al., 2018) or general purpose knowledge bases (Ren and Q. Zhu, 2021).

In this paper, we propose an unsupervised process to generate questions, answers and associated distractors in order to fine-tune and improve the performance of the state-of-the-art model UnifiedQA on unseen domains. This method, being unsupervised, needs no additional annotated domain specific data requiring only a set of unannotated sentences of the domain of interest from which the questions are created. Contrarily to most of the aforementioned works, our aim is not to train a new completely unsupervised model but rather to incorporate new information into an existing state-of-the-art model and thus to take advantage of the question-answering knowledge already learned.

We conduct our experiments on the SciQ dataset (Welbl et al., 2017). SciQ contains multiple-

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 732 - 738
May 22-27, 2022 ©2022 Association for Computational Linguistics

Question:
What type of organism is commonly used in preparation of foods such as cheese and yogurt?
(A) mesophilic organisms (B) protozoa
(C) gymnosperms (D) viruses

Support text:
Mesophiles grow best in moderate temperature, typically between 25°C and 40°C (77°F and 104°F). Mesophiles are often found living in or on the bodies of humans or other animals. The optimal growth temperature of many pathogenic mesophiles is 37°C (98°F), the normal human body temperature. Mesophilic organisms have important uses in food preparation, including cheese, yogurt, beer and wine.

Figure 1: Example of a question in SciQ. The answer in bold is the correct one.

choice questions (4 choices) featuring subjects centered around physics, biology and chemistry. An example of question can be found in Figure 1. We focus on the SciQ dataset because it has not yet been used for training UnifiedQA and it requires precise scientific knowledge. Furthermore, our experiments reveal that the direct application of UnifiedQA on the SciQ benchmark leads to a much lower performance than when fine-tuning it on the SciQ training set (see Section 4). Our objective in this work is to solve this gap between UnifiedQA and UnifiedQA fine-tuned on supervised data with the unsupervised question generation approach described in Section 2. We additionally test our method on two commonly used multiple choice question answering datasets: CommonsenseQA (Talmor et al., 2019) and QASC (Khot et al., 2020). These datasets contain questions with similar domains to SciQ even though the questions are slightly less specific. Furthermore, neither of them has been used during the initial training of UnifiedQA.

2 Question Generation Method

We propose a method for generating multiple-choice questions in order to fine-tune and improve UnifiedQA. This process is based on 3 steps. First, a set of sentences is being selected (Section 2.1) from which a generic question generation system is applied (Section 2.2). Then a number of distractors are added to each question (Section 2.3).

Dataset	Sentences	Questions
SciQ data	53 270	77 873
SciQ data (train only)	45 526	66 552
Wikipedia data	45 327	62 848

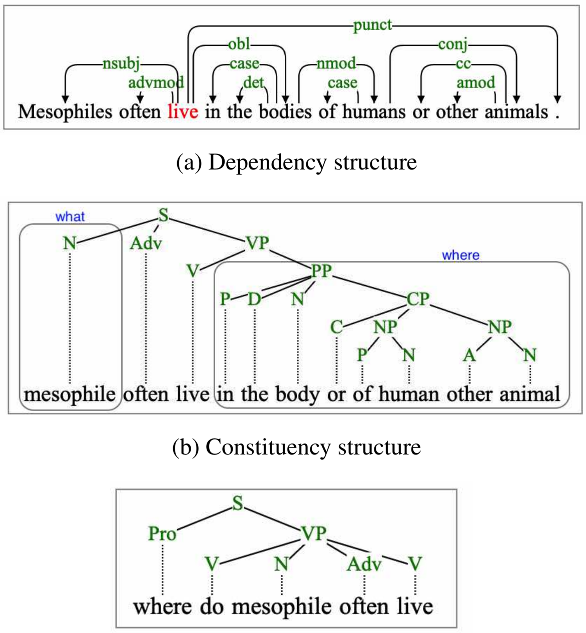
Table 1: Number of sentences selected for each of the datasets considered as well as the number of questions automatically generated from these sentences.

2.1 Sentence Selection

Our question generation method uses a set of unannotated sentences from which the questions will be generated. We compare three selection methods.

First, we consider a scenario where the application developer does not manually collect any sentence, but simply gives the name (or topic) of the target domain. In our case, the topics are “Physics”, “Biology” and “Chemistry” since these are the main domains in SciQ. A simple information retrieval strategy is then applied to automatically mine sentences from Wikipedia. We first compute a list of Wikipedia categories by recursively visiting all subcategories starting from the target topic names. The maximum recursion number is limited to 4. We then extract the summary (head paragraph of each Wikipedia article) for each of the articles matching the previously extracted categories and subcategories. We only keep articles with more than 800 average visitors per day for the last ten days (on April 27, 2021), resulting in 12 656 pages.

The two other selection methods extract sentences from SciQ itself and therefore are not entirely unsupervised but rather simulate a situation where we have access to unannotated texts that precisely describe the domains of interest such as a school book for example. The SciQ dataset includes a support paragraph for each question (see Figure 1). Pooled together, these support paragraphs provide us with a large dataset of texts about the domains of interest. We gather the paragraphs corresponding to all questions and split them into sentences to produce a large set of sentences that are no longer associated with any particular question but cover all the topics found in the questions. We compare two different setups. In the first one, we include all the sentences extracted from the train, validation and test sets thus simulating a perfect selection of sentences that cover all the knowledge expressed in the questions. Still, we only use the support paragraphs and not the annotated questions themselves. As compared to the classical


punct
obl
conj
nsubj
case
nmod
cc
advmod
det
case
amod
Mesophiles often live in the bodies of humans or other animals .
what
S
N
Adv
VP
V
where
PP
P
D
N
CP
C
NP
NP
P
N
A
N
mesophile
often live
in the body or of human other animal
S
Pro
VP
V
N
Adv
V
where do mesophile often live

(a) Dependency structure

(b) Constituency structure

(c) Generated question

Figure 2: Question generation process for a sentence similar to the one used to produce the question in Figure 3. A dependency parse (a) produced by Stanza is transformed into a constituency structure (b) in which two subtrees can be identified as the answers to two questions: *What* and *Where*. (c) shows the transformed constituency tree for the *Where* question.

Question:
What often is found living in or on the bodies of humans or other animals?
Right answer: mesophile

Random distractors:
(A) the most magnetic material in nature
(B) this energy
(C) climate

Refined distractors:
(A) carbohydrates
(B) small cell fragments called platelet
(C) echinoderm

Figure 3: Example of a synthetic question generated from the second sentence of the support paragraph in Figure 1 with a set of random distractors and with the set of refined ones.

supervised paradigm, this setting removes all annotation costs for the application developer, but it still requires to gather sentences that are deemed useful for the test set of interest. We then compare this setup with another one, where only the sentences from the train set are included. This scenario arguably meets more practical needs since it would suffice to gather sentences close to the domain of interest. The number of sentences for each dataset is presented in Table 1.

2.2 Questions Generation

The generation of questions from a sentence relies on the jsRealB text realizer (Lapalme, 2021) which generates an affirmative sentence from a constituent structure. It can also be parameterized to generate variations of the original sentence such as its negation, its passive form and different types of questions such as *who*, *what*, *when*, etc. The constituency structure of a sentence is most often created by a user or by a program from data. In this work, it is instead built from a Universal Dependency (UD) structure using a technique developed for *SR'19* (Lapalme, 2019). The UD structure of a sentence is the result of a dependency parse with Stanza (Qi et al., 2020). We thus have a pipeline composed of a neural dependency parser, followed by a program to create a constituency structure used as input for a text realizer, both in JavaScript. Used without modification, this would create a *complex* echo program for the original affirmative sentence, but by changing parameters, its output can vary.

In order to create questions from a single constituency structure, jsRealB uses the *classical* grammar transformations: for a *who* question, it removes the subject (i.e. the first noun phrase before the verb phrase), for a *what* question, it removes the subject or the direct object (i.e. the first noun phrase within the verb phrase); for other types of questions (*when*,*where*) it removes the first prepositional phrase within the verb phrase. Depending on the preposition, the question will be a *when* or a *where*. Note that the *removed* part becomes the answer to the question.

In order to determine which questions are appropriate for a given sentence, we examine the dependency structure of the original sentence and check if it contains the required part to be removed before parameterizing the realization. The generated questions are then filtered to remove any question for which the answer is composed of a single stopword. Table 1 shows the number of questions generated for each dataset. An example of a synthetic question is shown in Figure 3.

2.3 Distractors Selection

Since SciQ is a multiple-choice dataset, we must add distractors to each question we generate, to match the format of SciQ. A simple solution to this problem is to select random distractors among answers to other similar questions generated from the dataset of sentences we gathered. Obviously, selecting random distractors may lead to a fine-tuning dataset that is too easy to solve. Therefore, we propose another strategy that selects hard distractors for each question. To do so, starting from our synthetic dataset with random distractors, we fine-tune RoBERTa (Liu et al., 2019) using the standard method of training for multiple choices question answering. Each pair question/choice is fed to RoBERTa and the embedding corresponding to the first token ("[CLS]") is given to a linear layer to produce a single scalar score for each choice. The scores corresponding to every choice for a given question are then compared to each other by a softmax and a cross-entropy loss. With this method, RoBERTa is trained to score a possible answer for a given question, based on whether or not it is a credible answer to that question. For each question, we then randomly select a number of candidate distractors from the answers to other questions and we use our trained RoBERTa to score each of these candidates. The 3 candidates with the highest scores (and thus the most credible answers) are selected. The idea is that during this first training, RoBERTa will learn a large amount of simplistic logic. For example, because of the initial random selection of distractors, it is highly unlikely that even one of the distractors will be close enough to the question's semantic field. Furthermore, a lot distractors have an incorrect grammar (eg: a distractor might be plural when the question expects a singular). Therefore, in this initial training, RoBERTa might learn to isolate the answer with a corresponding semantic field or the one with correct grammar. The re-selection then minimizes the amount of trivial distractors and models trained on this new refined dataset will have to focus on deeper and more meaningful relations between the questions and the answers. The process is better shown in Figure 4, and an example of refined distractors can be found in Figure 3.

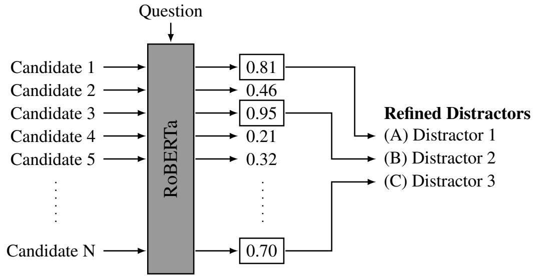


Figure 4: Description of the distractor refining method. RoBERTa scores each candidate distractor with regard to the question and the best 3 are selected to become the new refined distractors.

The number of scored candidate distractors is an hyper-parameter. A small number of candidates may result in a situation where none of the candidates are credible enough, while a large number requires more computation time, since the score of each candidate for every question needs to be computed, and has a higher risk of proposing multiple valid answers. In our experiments, we use a number of 64 candidates in order to limit computation time.

3 Training and Implementation Details

To refine distractors, we use the "Large" version of RoBERTa and all models are trained for 4 epochs and a learning rate of 1×10^{-5}. These hyper-parameters are chosen based on previous experiments with RoBERTa on other multiple-choice datasets. The final UnifiedQA fine-tuning is done using the same multiple choices question answering setup as the one used in the original UnifiedQA paper (Khashabi et al., 2020). We use the "Large" version of UnifiedQA and all the models are trained for 4 epochs using Adafactor and a learning rate of 1×10^{-5}. The learning rate is loosely tuned to get the best performance on the validation set during the supervised training of UnifiedQA. We use the Hugging Face pytorch-transformers (Wolf et al., 2020) library for model implementation. Experiments presented in this paper were carried out using the Grid'5000 testbed (Balouek et al., 2013), supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see `https://www.grid5000.fr`).

4 Results

Accuracy results in Table 2 have a 95% Wald confidence interval of $\pm 2.8\%$. The first row of Table 2 presents the accuracy results of a vanilla UnifiedQA large model on SciQ. The second line shows the accuracy when UnifiedQA is fine-tuned over the full training corpus. Our objective is thus to get as close as possible to this accuracy score using only unsupervised methods. The results using Wikipedia are the only ones that are unsupervised and therefore are the ones directly comparable to UnifiedQA with no fine-tuning or other unsupervised methods. The other results serve to illustrate what could be obtain with a tighter selection of sentences.

Model	**Dev**	**Test**
UnifiedQA (no fine-tuning)	64.6	63.4
UnifiedQA (supervised)	78.7	78.7
Unsupervised - Random distractors		
SciQ data	71.3	70.8
SciQ data (train only)	70.9	70.1
Wikipedia data	68.3	67.5
Unsupervised - Refined distractors		
SciQ data	75.4	74.2
SciQ data (train only)	73.1	72.4
Wikipedia data	70.6	69.4

Table 2: Accuracy on SciQ by UnifiedQA fine-tuned on our synthetic datasets. “SciQ data” refers to the questions generated using the support paragraphs in SciQ while “Wikipedia data” refers to questions generated using sentences harvested from Wikipedia. All scores are averaged over 3 independent runs (including the complete question generation process and the final UnifiedQA fine-tuning).

Fine-tuning UnifiedQA on synthetic questions with random distractors improves the results as compared to the baseline and, as expected, the closer the unlabeled sentences are to the topics of the questions, the better is the accuracy. Hence, generating questions from only the train set of SciQ gives performances that are comparable but slightly lower to the ones obtained from the combined train, dev and test set of SciQ. Finally, questions selected from Wikipedia also improve the results, despite being loosely related to the target test corpus. Our distractor selection method further boosts the accuracy results in all setups. This suggests that a careful selection of distractors is important, and that the hard selection criterion used here seems adequate in our context.

The results for CommonsenseQA and QASC using the same selection of sentences from Wikipedia are reported in table 3. Overall, we obtain similar results to SciQ with a large improvement of performances when generating questions and a further boost with refined distractors. However compared to SciQ, the improvement brought by the distractor refining process is less significant. This could be partly explained by the fact that the distractors in the original QASC and CommonsenseQA datasets are overall easier and therefore it is less advantageous for a model to be trained on harder questions.

Model	**CQA**	**QASC**
UnifiedQA (no fine-tuning)	60.9	44.5
UnifiedQA (supervised)	74.3	61.0
Wikipedia data (Random)	64.9	57.2
Wikipedia data (Refined)	65.1	59.4

Table 3: Accuracy results obtained on the dev set of CommonsenseQA and QASC when fine-tuning UnifiedQA using data from Wikipedia.

5 Conclusion

In this work, we proposed a multiple-choice question generation method that can be used to fine-tune the state-of-the-art UnifiedQA model and improve its performance on an unseen and out of domain dataset. Our contributions are:

- We have shown that simple unsupervised methods could be used to finetune existing multipurpose question answering models (in our case UnifiedQA) to new datasets or domains.
- We propose a novel distractor refining method able to select harder distractors for a given generated question and show its superiority compared to a random selection.

Future work includes comparing our method to other question generation methods (including supervised methods: Liu et al. (2020), Puri et al. (2020)) in order to assess the effect of both the generation method and the questions quality on the final performances of our models. Also, we will further compare different variations of our question generation and distractor refining methods in order to more thoroughly understand the effect of hyper-parameters such as the number of candidate distractors.

References

Daniel Balouek, Alexandra Carpen Amarie, Ghislain Charrier, Frédéric Desprez, Emmanuel Jeannot, Emmanuel Jeanvoine, Adrien Lèbre, David Margery, Nicolas Niclausse, Lucas Nussbaum, Olivier Richard, Christian Pérez, Flavien Quesnel, Cyril Rohr, and Luc Sarzyniec. 2013. Adding virtualization capabilities to the Grid’5000 testbed. In Ivan I. Ivanov, Marten van Sinderen, Frank Leymann, and Tony

Shan, editors, *Cloud Computing and Services Science*, volume 367 of *Communications in Computer and Information Science*, pages 3–20. Springer International Publishing.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186. Association for Computational Linguistics.

Alexander Fabbri, Patrick Ng, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. Template-based question generation from retrieved sentences for improved unsupervised question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4508–4513, Online. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. Qasc: A dataset for question answering via sentence composition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8082–8090.

Guy Lapalme. 2019. Realizing Universal Dependencies structures using a symbolic approach. In *The Second Multilingual Surface Realisation Shared Task (SR'19): Overview and Evaluation Results. In Proceedings of the 2nd Workshop on Multilingual Surface Realisation (MSR), (EMNLP-2019)*, page 8 pages, Hong-Kong. ACL.

Guy Lapalme. 2021. The jsRealB text realizer: Organization and use cases. (arXiv:2012.15425).

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910, Florence, Italy. Association for Computational Linguistics.

Zhongli Li, Wenhui Wang, Li Dong, Furu Wei, and Ke Xu. 2020. Harvesting and refining question-answer pairs for unsupervised QA. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6719–6728, Online. Association for Computational Linguistics.

Chen Liang, Xiao Yang, Neisarg Dave, Drew Wham, Bart Pursel, and C. Lee Giles. 2018. Distractor generation for multiple choice questions using learning to rank. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 284–290, New Orleans, Louisiana. Association for Computational Linguistics.

Bang Liu, Haojie Wei, Di Niu, Haolan Chen, and Yancheng He. 2020. Asking questions the human way: Scalable question-answer generation from text corpus. In *Proceedings of The Web Conference 2020*, pages 2032–2043.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826, Online. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

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

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

Siyu Ren and Kenny Q. Zhu. 2021. Knowledge-driven distractor generation for cloze-style multiple choice questions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5):4339–4347.

Keisuke Sakaguchi, Yuki Arase, and Mamoru Komachi. 2013. Discriminative approach to fill-in-the-blank quiz generation for language learners. In *Proceedings of the 51st Annual Meeting of the Association* *for Computational Linguistics (Volume 2: Short Papers)*, pages 238–242, Sofia, Bulgaria. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, Copenhagen, Denmark. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Can a Transformer Pass the Wug Test? Tuning Copying Bias in Neural Morphological Inflection Models

Ling Liu and **Mans Hulden**
University of Colorado
first.last@colorado.edu

Abstract

Deep learning sequence models have been successful with morphological inflection generation. The SIGMORPHON shared task results in the past several years indicate that such models can perform well, but only if the training data covers a good amount of different lemmata, or if the lemmata to be inflected at test time have also been seen in training, as has indeed been largely the case in these tasks. Surprisingly, we find that standard models such as the Transformer almost completely fail at generalizing inflection patterns when trained on a limited number of lemmata and asked to inflect previously unseen lemmata—i.e. under "wug test"-like circumstances. This is true even though the actual number of training examples is very large. While established data augmentation techniques can be employed to alleviate this shortcoming by introducing a copying bias through hallucinating synthetic new word forms using the alphabet in the language at hand, our experiment results show that, to be more effective, the hallucination process needs to pay attention to substrings of syllable-like length rather than individual characters.[1]

1 Introduction

The Transformer model has delivered convincing results in many different tasks related to word-formation and analysis (Vylomova et al., 2020; Moeller et al., 2020, 2021; Liu, 2021). Especially on inflection tasks, where an input lemma such as `dog`, and input inflectional features such as {`N,PL`}, are expected to produce an output such as `dogs`, the model has shown to be particularly adept at generalizing patterns (Vylomova et al., 2020; Liu and Hulden, 2020a,b; Wu et al., 2021). However, we have discovered that this is only true if the training data covers a diversity of lemmata or *some* variant of the input lemma to be inflected has been witnessed during training. In a "wug test" (Berko, 1958) setting where the witnessed lemmata are usually limited and a previously unseen lemma—like `wug`—is to be inflected in some way, we find that the Transformer almost completely fails to generalize inflection patterns, despite abundant inflected forms for training. It has been noted earlier that neural sequence-to-sequence models are apt to perform poorly for morphological inflection if they have been exposed to little training data and data augmentation can be leveraged to alleviate the problem (Cotterell et al., 2017, 2018; Kann and Schütze, 2017; Liu and Hulden, 2021). Our starting point is our observation that the poor "wug test" performance is maintained even with abundant training inflected forms.

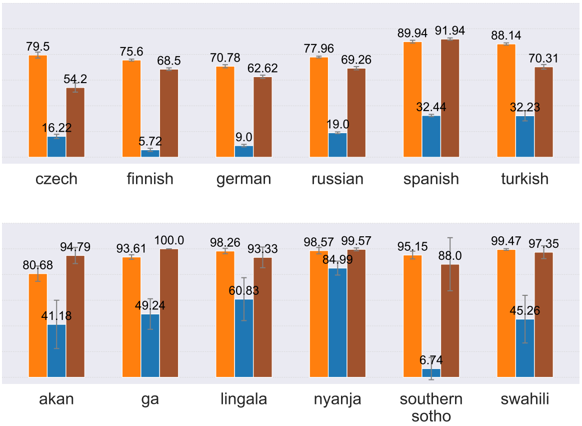


Figure 1: Transformer performance in the **common-practice** setting (left), "**wug test**"-like setting (middle), and ‘**‘wug test”-like setting with our best data hallucination** method (right)

In our study, we show three main results. (1) We demonstrate that, even if trained with relatively large amounts of inflected forms, a Transformer model of the kind that has been very successful at recent shared tasks largely fails to generalize inflection patterns if it has not been exposed during training to a variety of lemmata or any lemmata in

[1] The code and data are available at https://github.com/LINGuistLIU/transformer-wug-test.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 739 - 749
May 22-27, 2022 ©2022 Association for Computational Linguistics

the test set. This is true even for datasets where all words inflect in the same way—i.e. there are no inflectional classes or allomorphs of morphemes, as is found in the low-resource Niger-Congo language datasets used in SIGMORPHON 2020 shared task (Vylomova et al., 2020). (2) We show that simply exposing the model to uninflected lemmata in the test set—without providing a single inflected form—allows the model to dramatically improve its performance when inflecting such lemmata. (3) Further, we investigate several strategies that avoid leveraging test set lemmata. We show that when inducing a copy bias in the model by hallucinating new lemmata, or by hallucinating new inflected forms, the method of hallucination is much more effective if it is sensitive to substrings of syllable-like length rather than individual characters or stems. Our best models achieve substantial improvement upon earlier state-of-the-art data hallucination methods (Silfverberg et al., 2017; Anastasopoulos and Neubig, 2019).

2 Data

2018-languages We use six languages from the CoNLL-SIGMORPHON 2018 shared task 1 medium setting, where each language has 1,000 `(LEMMA, TARGET TAGS, TARGET FORM)` triples for training (Cotterell et al., 2018). The six languages, Czech, Finnish, German, Russian, Spanish and Turkish, are selected to represent the diversity of language typology and morphological inflection challenges. Though there are only 1,000 training triples, they cover a fair number of lemmata as each lemma appears only once or twice, an amount very hard to obtain for really low-resource languages. In the original shared task, between 2% and 27% of the lemmata in the dev and test sets are also found in the training set.

To prepare training data for the "wug test"-like circumstance, we select the UniMorph (Kirov et al., 2018) paradigms for the first 100 most frequent lexemes found in Wikipedia text,[2] which are not included in the 2018 shared task 1 dev and test sets. The shared task dev and test sets are used for validation and evaluation without any change. The 100 full inflection tables give us over 1,000 (for Czech, German and Russian) or over 7,000 (for Finnish, Spanish and Turkish) training triples.

Niger-Congo languages In addition, we use six Niger-Congo languages from SIGMORPHON 2020 shared task 0 (Vylomova et al., 2020): Akan, Ga, Lingala, Nyanja, Southern Sotho and Swahili. These languages are low-resource, but the dataset only contains very regular inflections. In the original shared task data split, The overlap between the lemmata in the dev and test sets and those in the training set is 100%. The number of paradigms which we can obtain by combining the training, dev and test sets of this dataset is around 100 for Akan, Ga and Swahili, 227 for Nyanja, 57 for Lingala and only 26 for Southern Sotho.

For the "wug test", we divide the inflection tables reconstructed from this dataset into a 7:1:2 train-dev-test split, i.e. we use the same ratio as the shared task, but the division is by inflection tables rather than lemma-tag-form triples, to ensure that the lemmata used for validation and test are disjoint from those for training. We provide details on the data statistics in Appendix A for reference.

3 Experiments

Inflection model The Transformer (Vaswani et al., 2017) is the seq2seq architecture which produces the current state-of-the-art result on the morphological inflection task (Vylomova et al., 2020; Liu and Hulden, 2020a,b; Wu et al., 2021). It takes the lemma and target tag(s) as input and predicts the target form character by character. Our experiments use the Transformer implemented in fairseq (Ott et al., 2019) and adopt the same hyperparameters as Liu and Hulden (2020a). [3]

Evaluation metric The evaluation metric is accuracy. For the original shared task data and experiments on 2018 languages, we train five inflection models each with a different random initialization and report the average accuracy with standard deviation. Due to data scarcity, for Niger-Congo languages at the "wug test"-like setting, we perform a 5-fold cross-validation and report the average accuracy and the standard deviation.

Common-practice test and "wug test" We first compare the performance of the Transformer in the common-practice setting and the "wug test"-like setting. The "common practice" is represented by

[2] We also experimented with using 100 random UniMorph lexemes, and did not find substantial difference between using random ones and the most frequent ones.

[3] We also conducted experiments with the encoder-decoder with hard monotonic attention model (Wu and Cotterell, 2019), but found the same conclusion as for the Transformer model. Experiments on the hard monotonic model is provided in Appendix C for reference.

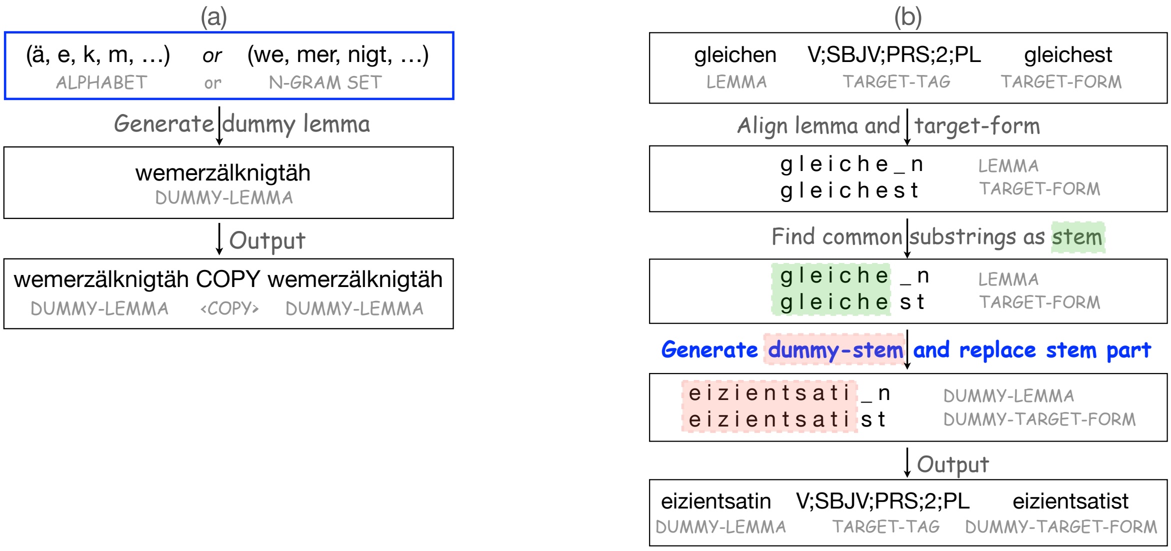

(a)
(ä, e, k, m, ...) or (we, mer, nigt, ...)
ALPHABET or N-GRAM SET
Generate dummy lemma
wemerzälknigtäh
DUMMY-LEMMA
Output
wemerzälknigtäh COPY wemerzälknigtäh
DUMMY-LEMMA <COPY> DUMMY-LEMMA
(b)
gleichen V;SBJV;PRS;2;PL gleichest
LEMMA TARGET-TAG TARGET-FORM
Align lemma and target-form
g l e i c h e _ n LEMMA
g l e i c h e s t TARGET-FORM
Find common substrings as stem
g l e i c h e _ n LEMMA
g l e i c h e s t TARGET-FORM
Generate dummy-stem and replace stem part
e i z i e n t s a t i _ n DUMMY-LEMMA
e i z i e n t s a t i s t DUMMY-TARGET-FORM
Output
eizientsatin V;SBJV;PRS;2;PL eizientsatist
DUMMY-LEMMA TARGET-TAG DUMMY-TARGET-FORM

Figure 2: (a) Dummy lemma generation with a German example. *+copy-2k-char* generates random strings by uniformly sampling from the alphabet, while *+copy-2k-substr* samples from the set of 2-, 3- and 4-grams; (b) Data hallucination with a German example. *+hall-2k-substr* is different from *+hall-2k-char* in how the dummy-stem is generated.

previous years' shared tasks and related work (Cotterell et al., 2016, 2017, 2018; McCarthy et al., 2019; Vylomova et al., 2020); here the training data usually covers a fair number of lemmata and there is overlap between lemmata in the training and test sets. We use the shared task data to represent the common-practice setting. In the "wug test" setting, we control the number of lemmata for training but not inflected forms (as explained in Section 2) and the lemmata to be inflected are always previously unseen. To our surprise, the performance of the Transformer at the "wug test"-like setting is very poor despite the large amount of training triples for 2018-languages or the very regular and straightforward inflection for Niger-Congo languages. The performance is dramatically inferior to the common-practice setting, even when the number of training triples is seven times larger for Finnish, Spanish and Turkish (see Figure 1).

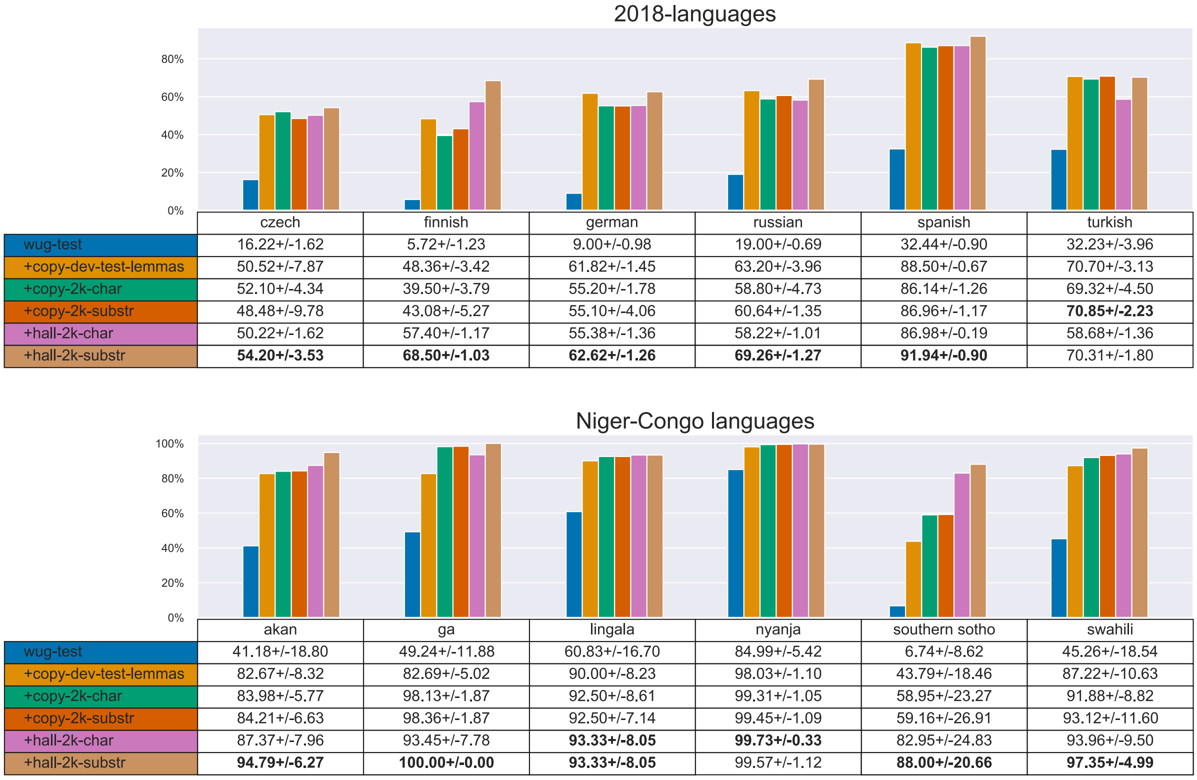
We hypothesize four reasons for the poor performance of the model under the "wug test"-like circumstance: (1) missing copy bias regarding the entire stem, i.e. the model can't copy a stem *abcde* if that exact stem has never been seen during training, (2) missing copy bias on individual letters, i.e. the model can't copy letter *a* if the letter is underrepresented in training, (3) missing copy bias on subsequences of letters, i.e. the model can't copy sequence *ab* if the sequence is underrepresented in training, (4) some combination of all the factors above. To test these hypotheses, we conduct five experiments designed to help the model learn to copy with different biases by adding to the training set for each language 2,000[4] dummy data points generated in five different ways, explained below.

+copy-dev-test-lemmas In order to test the first hypothesis that the model does not learn to copy parts of a stem it has not seen at the training stage, we augment the training data for each language by adding to it the lemmata in its development and test sets with a special tag `COPY`. In other words, 2,000 `(LEMMA, COPY, LEMMA)` triples are added to the initial "wug test" training set for each language.

+copy-2k-char* and *+copy-2k-substr Previous work found that adding random strings can help seq2seq models learn a copy bias and thus improve the performance when the training data is limited (Kann and Schütze, 2017). We adopt a similar method to augment the training data with dummy lemmata generated by the process shown in Figure 2 (a). The *+copy-2k-char* method takes as input the alphabet created by collecting characters in the language's training set.

Considering that a natural linguistic sub-unit of a word is a syllable, we propose to use sub-

[4] The choice of 2,000 is in order to match the augmentation size of *+copy-dev-test-lemmas* method for 2018-languages. We did not try to tune for the best data augmentation size. Appendix B provides plots of data augmentation size comparison, where we found no consistent difference in all the languages.



	czech	finnish	german	russian	spanish	turkish
wug-test	16.22+/-1.62	5.72+/-1.23	9.00+/-0.98	19.00+/-0.69	32.44+/-0.90	32.23+/-3.96
+copy-dev-test-lemmas	50.52+/-7.87	48.36+/-3.42	61.82+/-1.45	63.20+/-3.96	88.50+/-0.67	70.70+/-3.13
+copy-2k-char	52.10+/-4.34	39.50+/-3.79	55.20+/-1.78	58.80+/-4.73	86.14+/-1.26	69.32+/-4.50
+copy-2k-substr	48.48+/-9.78	43.08+/-5.27	55.10+/-4.06	60.64+/-1.35	86.96+/-1.17	**70.85+/-2.23**
+hall-2k-char	50.22+/-1.62	57.40+/-1.17	55.38+/-1.36	58.22+/-1.01	86.98+/-0.19	58.68+/-1.36
+hall-2k-substr	**54.20+/-3.53**	**68.50+/-1.03**	**62.62+/-1.26**	**69.26+/-1.27**	**91.94+/-0.90**	70.31+/-1.80

	akan	ga	lingala	nyanja	southern sotho	swahili
wug-test	41.18+/-18.80	49.24+/-11.88	60.83+/-16.70	84.99+/-5.42	6.74+/-8.62	45.26+/-18.54
+copy-dev-test-lemmas	82.67+/-8.32	82.69+/-5.02	90.00+/-8.23	98.03+/-1.10	43.79+/-18.46	87.22+/-10.63
+copy-2k-char	83.98+/-5.77	98.13+/-1.87	92.50+/-8.61	99.31+/-1.05	58.95+/-23.27	91.88+/-8.82
+copy-2k-substr	84.21+/-6.63	98.36+/-1.87	92.50+/-7.14	99.45+/-1.09	59.16+/-26.91	93.12+/-11.60
+hall-2k-char	87.37+/-7.96	93.45+/-7.78	**93.33+/-8.05**	**99.73+/-0.33**	82.95+/-24.83	93.96+/-9.50
+hall-2k-substr	**94.79+/-6.27**	**100.00+/-0.00**	**93.33+/-8.05**	99.57+/-1.12	**88.00+/-20.66**	**97.35+/-4.99**

Figure 3: "Wug test" results. *+copy-2k-char* adds random strings generated with the alphabet. *+copy-2k-substr* adds random strings generated with the n-gram set. *+hall-2k-char* adds data hallucinated with the method by Anastasopoulos and Neubig (2019). *+hall-2k-substr* adds data hallucinated with our method.

strings of syllable-like length for the *+copy-2k-substr* method. The input of this method is the set of bigrams, trigrams and four-grams from the language's training data. For both methods, we generate the dummy lemma by uniformly sampling from the input and concatenating the sampled items to a random length between the minimum and maximum word length we see in the training data. The output of the dummy lemma generation process is a triple of a dummy lemma, a special symbol `COPY` and the dummy lemma, which is added to the initial "wug test" training set for data augmentation.

+hall-2k-char* and *+hall-2k-substr The dummy lemma generation methods do not leverage knowledge about word structure which can be inferred from the training data. Silfverberg et al. (2017) found that it is very effective to augment training data in low-resource situations with a data hallucination approach by replacing a hypothesized stem of the training triples with a random string. Anastasopoulos and Neubig (2019) improves this data hallucination method by taking into discontinuous stems into consideration as well; this is the best data hallucination method so far. We conduct the *+hall-2k-char* experiment by augmenting the initial "wug test" training set with dummy data generated with Anastasopoulos and Neubig (2019)'s method. The implementation from SIGMORPHON 2020 shared task 0 baseline is used.

In addition, we propose to generate the dummy stem by uniformly sampling from substrings of syllable-like length, i.e. the bigram, trigram and four-gram set. This experiment is referred to as *+hall-2k-substr*. Specifically, both data hallucination methods (illustrated in Figure 2 (b)) take as input a triple from the training set, aligns the lemma and the target form with the alignment method from SIGMORPHON 2016 shared task baseline (Cotterell et al., 2016), finds the common substrings between the lemma and the target form as the stem, replaces the stem with a dummy stem, and outputs a dummy triple which is adopted for data augmentation. Our proposed method is different from Anastasopoulos and Neubig (2019)'s method at the dummy stem generation step in two main aspects: (1) Instead of sampling from the alphabet, we sample from the set of bigrams, trigrams and four-grams. (2) Instead of forcing the dummy stem to be of the same length as the stem to be replaced, we only constrain the minimum and maximum length of the stem based on the training data. In addition, for discontinuous stems, we only replace the first part of the stem.[5]

4 Results and discussion

"Wug test" with data augmentation Figure 3 shows results for the "wug test"-like setting and results after augmenting the initial training set with different methods. Every language sees a substantial improvement with data augmentation, indicating that the Transformer model in the vanilla "wug test" circumstance will not learn a copy bias well.

The substring-based data hallucination we propose, *+hall-2k-substr*, achieves accuracies which are substantially higher than other methods for most languages. For Turkish and Nyanja, *+hall-2k-substr* is lower than the best performance, but the difference is not obvious. For Lingala, *+hall-2k-substr* has the same best performance as *+hall-2k-char*. The consistent advantage of *+hall-2k-substr* implies that substrings of syllable-like length is more helpful than individual characters for data hallucination. It also provides support to the fourth hypothesis we made in section 3 that the poor performance of the Transformer in the vanilla "wug test"-like setting is due to a combination of factors including missing copying bias for letters, subsequences of letters and even entire stems.

Common practice vs "wug test" Figure 1 plots the Transformer accuracies with standard deviations in the common-practice setting, vanilla "wug test"-like setting, and "wug test"-like setting with data augmentation by the substring-based data hallucination methods (*+hall-2k-substr*). Though data augmentation can improve the model's performance for a "wug test", results are still inferior to the common practice setting without any data augmentation for most languages, especially the morphologically challenging 2018 CoNLL-SIGMORPHON languages.

5 Conclusion

In this work, we examine limiting the number of training lemmata and keeping training lemmata disjoint from the evaluation sets in morphological inflection. By comparing the performance of the Transformer under the "wug test"-like circumstance with the common practice, we find that the common-practice setting where the training data covers a fair amount of lemmata and there is overlap of lemmata in training and evaluation, has obscured the difficulty of the task. We propose to augment the training data with substring-based data hallucination, and achieve substantial improvement over previous data hallucination methods.

Considering the findings in this paper, we suggest that future experiments include evaluations on model performance using lemmata not found in the training set and use unique lemma counts rather than triple counts to document data set sizes.

[5] Using the first part only is for implementation simplicity in the current work. It should be adjusted for languages with a large number of discontinuous stems.

References

Roee Aharoni and Yoav Goldberg. 2017. Morphological inflection generation with hard monotonic attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2004–2015, Vancouver, Canada. Association for Computational Linguistics.

Roee Aharoni, Yoav Goldberg, and Yonatan Belinkov. 2016. Improving sequence to sequence learning for morphological inflection generation: The BIU-MIT systems for the SIGMORPHON 2016 shared task for morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 41–48. Association for Computational Linguistics.

Antonios Anastasopoulos and Graham Neubig. 2019. Pushing the limits of low-resource morphological inflection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 984–996, Hong Kong, China. Association for Computational Linguistics.

Jean Berko. 1958. The child's learning of English morphology. *Word*, 14(2-3):150–177.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017.

CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 1–30, Vancouver. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The SIGMORPHON 2016 shared Task—Morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22, Berlin, Germany. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2017. Unlabeled data for morphological generation with character-based sequence-to-sequence models. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 76–81, Copenhagen, Denmark. Association for Computational Linguistics.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sabrina J. Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: Universal Morphology. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Ling Liu. 2021. Computational morphology with neural network approaches. *arXiv preprint arXiv:2105.09404*.

Ling Liu and Mans Hulden. 2020a. Analogy models for neural word inflection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2861–2878, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ling Liu and Mans Hulden. 2020b. Leveraging principal parts for morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 153–161, Online. Association for Computational Linguistics.

Ling Liu and Mans Hulden. 2021. Backtranslation in neural morphological inflection. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 81–88, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Peter Makarov and Simon Clematide. 2018a. Imitation learning for neural morphological string transduction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2877–2882. Association for Computational Linguistics.

Peter Makarov and Simon Clematide. 2018b. Neural transition-based string transduction for limited-resource setting in morphology. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 83–93. Association for Computational Linguistics.

Peter Makarov and Simon Clematide. 2018c. UZH at CoNLL–SIGMORPHON 2018 shared task on universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 69–75, Brussels. Association for Computational Linguistics.

Peter Makarov, Tatiana Ruzsics, and Simon Clematide. 2017. Align and copy: UZH at SIGMORPHON 2017 shared task for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 49–57, Vancouver. Association for Computational Linguistics.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sabrina J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy. Association for Computational Linguistics.

Sarah Moeller, Ling Liu, and Mans Hulden. 2021. To POS tag or not to POS tag: The impact of POS tags on morphological learning in low-resource settings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 966–978, Online. Association for Computational Linguistics.

Sarah Moeller, Ling Liu, Changbing Yang, Katharina Kann, and Mans Hulden. 2020. IGT2P: From interlinear glossed texts to paradigms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5251–5262, Online. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. In *Proceedings of the*

CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection, pages 90–99, Vancouver. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Ekaterina Vylomova, Jennifer White, Elizabeth Salesky, Sabrina J. Mielke, Shijie Wu, Edoardo Maria Ponti, Rowan Hall Maudslay, Ran Zmigrod, Josef Valvoda, Svetlana Toldova, Francis Tyers, Elena Klyachko, Ilya Yegorov, Natalia Krizhanovsky, Paula Czarnowska, Irene Nikkarinen, Andrew Krizhanovsky, Tiago Pimentel, Lucas Torroba Hennigen, Christo Kirov, Garrett Nicolai, Adina Williams, Antonios Anastasopoulos, Hilaria Cruz, Eleanor Chodroff, Ryan Cotterell, Miikka Silfverberg, and Mans Hulden. 2020. SIGMORPHON 2020 shared task 0: Typologically diverse morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 1–39, Online. Association for Computational Linguistics.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1530–1537, Florence, Italy. Association for Computational Linguistics.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2021. Applying the transformer to character-level transduction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1901–1907, Online. Association for Computational Linguistics.

Shijie Wu, Pamela Shapiro, and Ryan Cotterell. 2018. Hard non-monotonic attention for character-level transduction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4425–4438, Brussels, Belgium. Association for Computational Linguistics.

A Data information

	triple-counts			lemma-counts			lemma-overlap (%)	
Language	train	dev	test	train	dev	test	dev-train	test-train
czech	1000	1000	1000	848	848	849	24.53	20.38
finnish	1000	1000	1000	985	983	987	2.34	3.04
german	1000	1000	1000	961	945	962	9.42	9.46
russian	1000	1000	1000	973	985	977	3.65	3.79
spanish	1000	1000	1000	906	902	922	15.74	16.49
turkish	906	928	912	764	802	779	26.06	26.57

Table 1: CoNLL-SIGMORPHON 2018 shared task 1 medium-size data information.

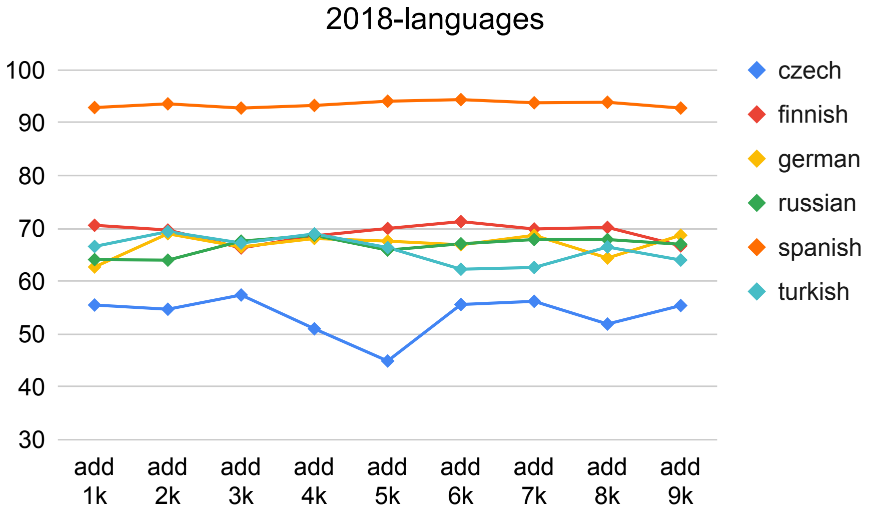
	triple-counts	lemma-counts	lemma-overlap (%)	
Language	train	train	dev-train	test-train
czech	1582	100	0	0
finnish	7136	100	0	0
german	1290	100	0	0
russian	1311	100	0	0
spanish	7132	100	0	0
turkish	7632	100	0	0

Table 2: Data information of the training set we create for 2018-languages. We use the same dev and test sets as CoNLL-SIGMORPHON 2018 shared task 1.

	triple-counts			lemma-counts			lemma-overlap (%)	
Language	train	dev	test	train	dev	test	dev-train	test-train
akan	2793	380	763	96	94	95	100.0	100.0
ga	607	79	169	95	59	80	100.0	100.0
lingala	159	23	46	57	23	34	100.0	100.0
nyanja	3031	429	853	227	199	226	100.0	100.0
southern sotho	345	50	99	26	24	25	100.0	100.0
swahili	3374	469	910	97	97	96	100.0	100.0

Table 3: Data information of Niger-Congo languages from SIGMORPHON 2020 shared task 0.

B Data augmentation size comparison



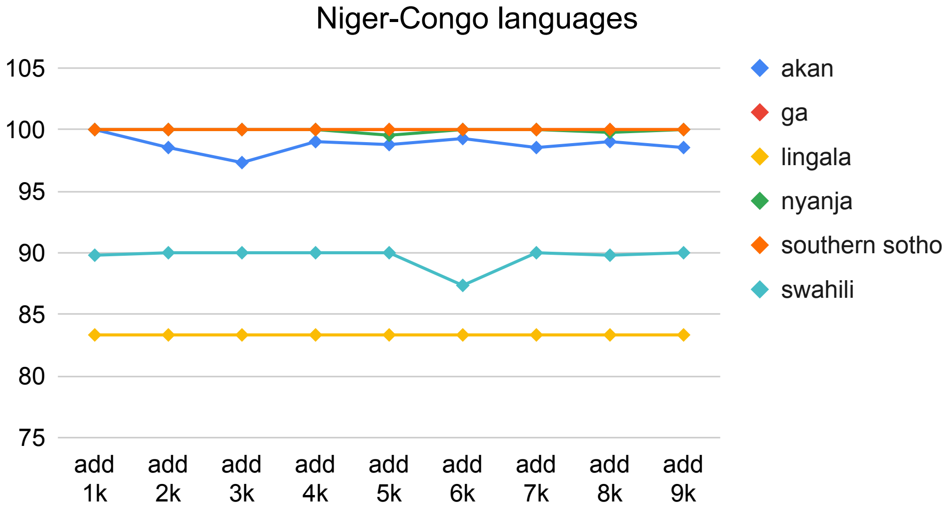


Figure 4: Performance on the dev set in "wug test" after **adding different amounts of dummy data** generated with our substring-based hallucination method.

C Performance of the encoder-decoder with hard monotonic attention model

Considering that the encoder-decoder with hard monotonic attention model (Aharoni et al., 2016; Aharoni and Goldberg, 2017; Makarov et al., 2017; Makarov and Clematide, 2018c,a,b; Wu et al., 2018; Wu and Cotterell, 2019) is designed for the morphological generation task and bias towards copying symbols in the input by leveraging edit actions, we evaluate the performance of the encoder-decoder with exact hard monotonic attention in the “wug test”-like circumstance as well in order to evaluate whether this deep learning model architecture catered to morphological generation is able to learn the generalization ability. We use the encoder-decoder with exact hard monotonic attention model proposed and implemented by Wu and Cotterell (2019).[6]

The performance of the encoder-decoder with exact hard monotonic attention model for the original shared task setup, the “wug test”-like setup with or without our best data hallucination augmentation is presented in Figure 5. Figure 6 provides detailed comparison between different data augmentation methods in the “wug test”-like experimental setup by the encoder-decoder with exact hard monotonic attention model. We observe that the encoder-decoder with exact hard monotonic attention model has the same limitation as the Transformer model pointed out in the previous section.

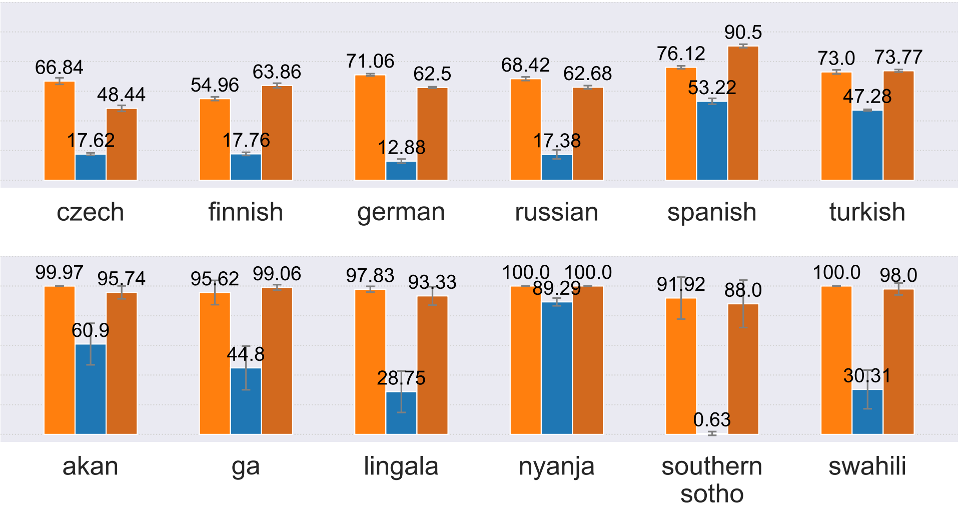
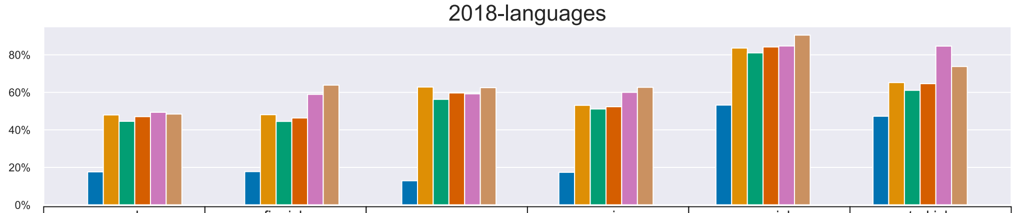
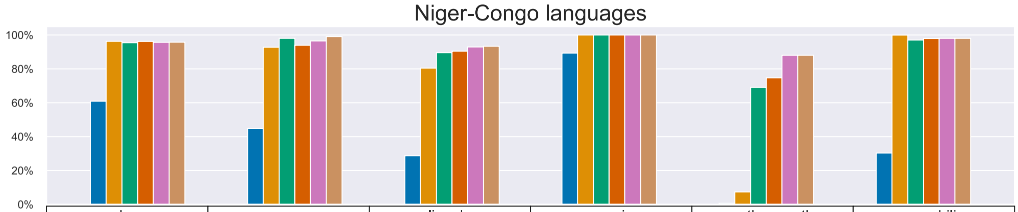


Figure 5: Performance of the encoder-decoder with exact hard monotonic attention model (Wu and Cotterell, 2019) in the **common-practice** setting (left), “**wug test**”-like setting (middle), and **“wug test”-like setting with our best data hallucination** method (right)

[6] https://github.com/shijie-wu/neural-transducer


2018-languages

	czech	finnish	german	russian	spanish	turkish
wug-test	17.62+/-0.79	17.76+/-1.07	12.88+/-1.40	17.38+/-3.05	53.22+/-1.99	47.28+/-0.51
+copy-dev-test-lemmas	47.96+/-2.24	48.10+/-1.37	**62.84+/-1.11**	53.08+/-1.79	83.62+/-1.21	65.22+/-1.40
+copy-2k-char	44.58+/-3.51	44.52+/-0.73	56.28+/-1.53	51.14+/-1.54	81.04+/-1.56	61.08+/-1.64
+copy-2k-substr	47.06+/-2.65	46.34+/-0.92	59.68+/-1.12	52.34+/-1.38	84.20+/-0.24	64.63+/-1.52
+hall-2k-char	**49.38+/-0.79**	58.92+/-0.98	59.20+/-1.31	60.00+/-0.51	84.72+/-0.67	**84.65+/-0.57**
+hall-2k-substr	48.44+/-2.03	**63.86+/-1.54**	62.50+/-0.49	**62.68+/-1.12**	**90.50+/-1.18**	73.77+/-0.85


Niger-Congo languages

	akan	ga	lingala	nyanja	southern sotho	swahili
wug-test	60.90+/-14.01	44.80+/-14.74	28.75+/-13.97	89.29+/-2.63	0.63+/-1.26	30.31+/-13.05
+copy-dev-test-lemmas	**96.25+/-2.35**	92.75+/-7.86	80.42+/-19.70	**100.00+/-0.00**	7.37+/-4.36	**99.94+/-0.08**
+copy-2k-char	95.46+/-3.94	98.01+/-2.66	89.58+/-7.45	**100.00+/-0.00**	69.05+/-29.05	97.02+/-4.95
+copy-2k-substr	96.20+/-4.05	93.92+/-3.54	90.42+/-5.83	**100.00+/-0.00**	74.74+/-17.93	97.92+/-3.96
+hall-2k-char	95.66+/-3.95	96.49+/-4.44	92.92+/-6.40	**100.00+/-0.00**	**88.00+/-16.00**	98.00+/-4.00
+hall-2k-substr	95.74+/-4.34	**99.06+/-1.87**	**93.33+/-6.24**	**100.00+/-0.00**	**88.00+/-16.00**	98.00+/-4.00

Figure 6: “Wug test” results by the encoder-decoder with exact hard monotonic attention model (Wu and Cotterell, 2019), with or without different data augmentation methods.

Probing the Robustness of Trained Metrics for Conversational Dialogue Systems

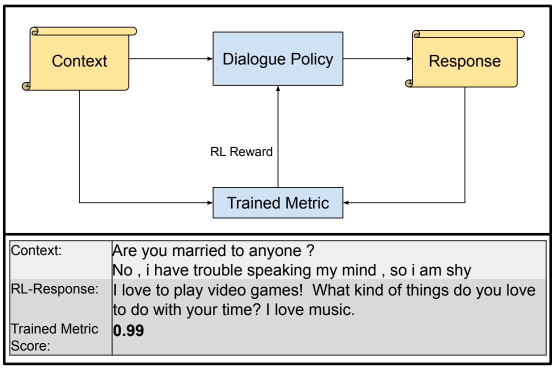
Jan Deriu, Don Tuggener, Pius von Däniken, Mark Cieliebak
Zurich University of Applied Sciences (ZHAW), Winterthur, Switzerland
deri@zhaw.ch

Abstract

This paper introduces an adversarial method to stress-test trained metrics to evaluate conversational dialogue systems. The method leverages Reinforcement Learning to find response strategies that elicit optimal scores from the trained metrics. We apply our method to test recently proposed trained metrics. We find that they all are susceptible to giving high scores to responses generated by relatively simple and obviously flawed strategies that our method converges on. For instance, simply copying parts of the conversation context to form a response yields competitive scores or even outperforms responses written by humans.

1 Introduction

One major issue in developing conversational dialogue systems is the significant efforts required for evaluation. This hinders rapid developments in this field because frequent evaluations are not possible or very expensive. The goal is to create automated methods for evaluating to increase efficiency. Unfortunately, methods such as BLEU (Papineni et al., 2002) have been shown to not be applicable to conversational dialogue systems (Liu et al., 2016). Following this observation, in recent years, the trend towards training methods for evaluating dialogue systems emerged (Lowe et al., 2017; Deriu and Cieliebak, 2019; Mehri and Eskenazi, 2020; Deriu et al., 2020). The models are trained to take as input a pair of context and candidate response, and output a numerical score that rates the candidate for the given context. These systems achieve high correlations to human judgments, which is very promising. Unfortunately, these systems have been shown to suffer from instabilities. (Sai et al., 2019) showed that small perturbations to the candidate response already confuse the trained metric. This work goes one step further: we propose a method that automatically finds strategies that elicit very high scores from the trained metric while being of obvious low quality. Our method can be applied to automatically test the robustness of trained metrics against adversarial strategies that exploit certain weaknesses of the trained metric.



Context:	Are you married to anyone ? No , i have trouble speaking my mind , so i am shy
RL-Response:	I love to play video games! What kind of things do you love to do with your time? I love music.
Trained Metric Score:	**0.99**

Figure 1: Overview of the process. It takes a context and an response generated by a dialogue policy and computes a score based on the trained metric. The score is then used as a reward to update the policy. In this example, the policy converges to a fixed response, which achieves an almost perfect score, although it is clearly a low-quality response. The policy always returns this response, regardless of the context, and the trained metric always scores it perfectly.

Our method uses a trained metric as a reward in a Reinforcement Learning setting, where we fine-tune a dialogue system to maximize the reward. Using this approach, the dialogue system converges towards a degenerate strategy that gets high rewards from the trained metric. It converges to three different degenerate types of strategies to which the policy converges in our experiments: the *Parrot*, the *Fixed Response*, and the *Pattern*. For each dataset and metric, an adversarial response is found, which belongs to one of the three strategy types. The responses generated from these strategies then achieve high scores on the metric. Even more, in most cases, the scores are higher than the scores achieved by human written responses. Figure 1 shows the pipeline. The dialogue policy receives a reward signal from the trained metric.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 750 - 761
May 22-27, 2022 ©2022 Association for Computational Linguistics

Over time, the policy converges to a fixed response, which objectively does not match the context but gets a near-perfect score on the trained metric. We release the code [1].

2 Related Work

Trained Metrics. In recent years the field of trained metrics gained traction after word-overlap methods have been shown to be unreliable (Liu et al., 2016). The first of these metrics is ADEM (Lowe et al., 2017), which takes as input a context, a reference, and the candidate response and returns a score. The main issue with ADEM is the reliance on references and annotated data (i.e., human ratings of responses), which are costly to obtain, and need to be redone for each domain. RUBER (Tao et al., 2018) extended ADEM by removing the reliance on annotated data for training. However, it still relies on a reference during inference. AutoJudge (Deriu and Cieliebak, 2019) removed the reliance on references, which allows the evaluation of multi-turn behavior of the dialogue system. However, AutoJudge still leverages annotated data for training. USR (Mehri and Eskenazi, 2020) is a trained metric that does not rely on either annotated data or any reference. It is trained in a completely unsupervised manner while still highly correlated to human judgment (0.4 Spearman Correlation). Similarly, MAUDE (Sinha et al., 2020) is trained as an unreferenced metric built to handle the online evaluation of dialogue systems.

Robustness of Trained Metrics. There is not yet much research on the robustness of trained metrics. Sai et al. (2019) evaluated the robustness of ADEM by corrupting the context in different ways. They show that by just removing punctuation, the scores of ADEM change, and in 64% of cases are superior to the scores given for the same response without removed punctuation. Other corruption mechanisms yielded similar results. Yeh et al. (2021) compared a large variety of automated metrics for dialogue system evaluation by comparing, e.g., turn- and dialogue-level correlation with human judgemnts and studying the impact of the dialogue length. They find that no single metric is robust against all alternations but see potential in ensembling different metrics. Novikova et al. (2017) investigate automated metrics in the task-oriented NLG domain and find that the metrics do not sufficiently reflect human ratings.

Algorithm 1: Advantage Actor-Critic Algorithm, where π_θ denotes the policy, c denotes the context, r the response generated by the policy, and s denotes the score by the automated metric, i.e., the reward.

```
while training do
    sample $c$ from pool of contexts;
    $r = \pi_\theta(c)$ generate response;
    $s = R(c, r)$ compute reward;
    fit action-value function $Q_\sigma$ i.e., $\mathcal{L}(\sigma) = \frac{1}{2}\sum_i \left\| R(c,r) + Q_(c', r') - Q_\sigma(c,r) \right\|$;
      compute the advantage $A(r,c) = R(r,c) - Q(c,r) + Q(c', r')$;
    $\theta = \theta + \alpha \bigtriangledown J_{RL}(\theta)$ fit policy;
end
```

3 Method

Our method applies a trained metric as a reward signal $R(c, r)$ to update a dialogue system $\pi(c)$ in a reinforcement learning setting, where c denotes the context and r the response. The dialogue system is trained by generating a response for a context, which is then scored by the automated metric. The dialogue system is then updated using the score as the reward. This process is repeated for different contexts. We use the Actor-Critic framework to optimize the policy (Sutton et al., 1999). See Algorithm 1 for an overview. The policy gradient is defined as $\bigtriangledown J_{RL}(\theta) = \bigtriangledown_\theta log\ \pi_\theta(r|c) * A(r, c)$, where $\pi_\theta(r|c)$ defines the probability of the generated response for the given context, and $A(c, r)$ the advantage function.

The learned policy depends on the reward function, i.e., the automated metric. If the reward function is susceptible to adversarial attacks, the policy will likely generate an objectively suboptimal solution, which is rated highly by the automated metric. Conversely, we expect the policy to improve the dialogue systems' responses if the automated metric is robust against adversarial examples.

4 Experimental Setup

4.1 Datasets

We perform the evaluation on three widely-used datasets in the dialogue modelling domain. Namely, Dailydialog (Li et al., 2017), Empathetic Dialogues (Rashkin et al., 2019), and PersonaChat (Zhang et al., 2018).

[1] `https://github.com/jderiu/metric-robustness`

Metric	Strategy	Response
PersonaChat		
ATT	Fixed	yea!!! 1!! 2!! 3!! * * * fucking fucking fucking * * [[[fucking * fucking *
BLM	Fixed	that sounds like a lot of fun. what do you like to do in your spare time?
MAUDE	Fixed	What kind of work do you have? What do you like to do in your free time?
USR Full	Parrot	-
USR MLM	Fixed	i am a stay at home mom and i am trying to figure out what i want to do with my life
USR Ret	Fixed	I love to be a musician. I love music. What kind of music do you listen to as a music lover
Dailydialog		
ATT	Fixed	! freaking out! one of these days! * * one * * freaking * * out! * even * * damn * * even damn
BLM	Fixed	that would be great! what do you do for a living, if you don't mind me asking?
MAUDE	Fixed	I hope it works out for you. What kind of car did you get?
USR Full	Pattern	i'm not sure if i'd like to [copy context tokens]. i'll let you know if i do.
USR MLM	Fixed	i am not sure if i am going to be able to go out of my way to get to know each other or not.
USR Ret	Parrot	-
Empathetic Dialogues		
ATT	Fixed	I know right? I felt SO SO ASHAmed of myself. I felt so embar assed.
BLM	Fixed	I'm so sorry to hear that. What happened, if you don't mind me asking?
MAUDE	Fixed	I wish I could go back in time and be a kid again. I miss those days.
USR Full	Pattern	i don't think it's [random context noun]. i'm sorry to hear that. what do you mean by that?
USR MLM	Fixed	I don't know what I'm going to do if it doesn't work out. I'm not sure what to do.
USR Ret	Parrot	-

Table 1: The strategies achieved for each metric and domain.

4.2 Metrics

We use various state-of-the-art automated metrics developed for evaluating conversational dialogue systems without reference, i.e., so-called unreferenced metrics.. These are metrics where no reference is needed, i.e. they only use the context and response to determine the score. They can be represented as a function $s = R(c, r)$, which rate the response r for a given context c.

We selected state-of-the-art trained metrics which achieve good correlations to human judgments to evaluate our approach—namely, USR (Mehri and Eskenazi, 2020), ATT (Gao et al., 2021), and MAUDE (Sinha et al., 2020). Additionally, we added the Blender language model score (BlenderLM) (Roller et al., 2020). For the ATT [2], MAUDE [3], and BlenderLM metrics [4], we use the out-of-the-box models provided by the respective authors. For the USR metric, we perform custom training on each dataset. Furthermore, we report the USR-retrieval (*USR Ret*), USR-masked-language-model *USR MLM*, and the USR-regression *USR Full* scores. Note that the *USR Full* is a combination of the *USR Ret* and *USR MLM* metric. More details can be found in Appendix A.

4.3 Strategies

For our approach, we use Blenderbot as our policy (Roller et al., 2020) since it is currently a state-of-the-art conversational dialogue system [5]. We use the validation set for each domain to perform reinforcement learning. This is to avoid the dialogue systems being fine-tuned on already seen data. We use the test set to evaluate the reward over the number of episodes. We perform the reinforcement learning for 15 epochs, where each epoch is composed of 500 updates. We noted from pre-experiments that this is enough for a dialogue system to converge to a degenerate strategy. We track the average reward achieved on the test set after each epoch. Each experiment is repeated 10 times since we expect the policy to converge to slightly different strategies in different runs. We select the repetition which achieved the highest score (i.e., reward) and use it to determine the strategy. We also experimented with automated strategy detection, see Appendix B.

5 Results

The policies typically converge towards one of the following three degenerate strategies.

Parrot. Here, the policy simply copies parts of the context into the response. Sometimes, it applies slight changes. For instance, it changes the pronouns from "you" to "I".

Fixed Response. Here, the policy converges on a fixed response which it returns regardless of the

[2] https://github.com/golsun/AdversarialTuringTest

[3] https://github.com/facebookresearch/online_dialog_eval

[4] https://huggingface.co/facebook/blenderbot-400M-distill

[5] Note that here we are referring to Blenderbot as a dialogue system. BLM is using the Blenderbot LM as a metric.

Dailydialog

	USR Ret	USR MLM	USR Full	ATT	MAUDE	BLM
BL	0.440	0.426	4.951	0.0002	0.664	0.096
HU	0.928	0.409	7.904	0.0006	0.898	0.183
Copy	0.998	0.811	9.429	0.0002	0.921	0.233
Fixed	-	**0.505**	-	**0.435**	**0.985**	**0.239**
Parrot	**0.998**	-	-	-	-	-
Pattern	-	-	**7.091**	-	-	-

Empathetic Dialogues

	USR Ret	USR MLM	USR Full	ATT	MAUDE	BLM
BL	0.935	0.298	7.645	0.001	0.820	0.087
HU	0.891	0.384	7.611	0.120	0.942	0.264
Copy	0.996	0.885	9.617	0.054	0.935	0.358
Fixed	-	**0.912**	-	**0.731**	**0.976**	**0.333**
Parrot	**0.994**	-	-	-	-	-
Pattern	-	-	**7.240**	-	-	-

PersonaChat

	USR Ret	USR MLM	USR Full	ATT	MAUDE	BLM
BL	0.847	0.185	6.797	0.0006	0.844	0.070
HU	0.927	0.267	7.512	0.0024	0.951	0.153
Copy	0.925	0.794	8.933	0.0001	0.898	0.223
Fixed	**0.977**	**0.852**	-	**0.813**	**0.933**	**0.250**
Parrot	-	-	**7.542**	-	-	-
Pattern	-	-	-	-	-	-

Table 2: Scores achieved by humans (HU), Blenderbot (BL) and the degenerate strategies with regard to the different metrics for each domain.

context.

Pattern. This is a mix between the *Parrot* and the *Fixed Response*. It creates a fixed template filled with parts of the context.

Table 1 shows the selected responses for each pair of domain and metric. For all metrics except *ATT*, the fixed response is composed of a grammatically correct sentence. Note that these responses are always returned by the fine-tuned dialogue system, regardless of the context.

5.1 Scores

Table 2 shows the main results. In almost all cases, the degenerated strategy outperforms the vanilla Blenderbot and humans with respect to the automated metric. The most striking example is the *ATT* metric, where the fixed response achieves scores by orders of magnitude better than the ones achieved by humans. For both *USR Ret* and *MAUDE*, the scores achieved by the fixed response are almost perfect, i.e., they are close to 1.0, which is the upper bound. Also, for *USR MLM*, the scores are significantly higher than the ones achieved by Blenderbot. Interestingly, the *USR FULL* seems to be more immune to the pattern that were found. However, even for *USR FULL*, the parrot strategy beats the humans by a significant margin in the *PersonaChat* domain.

Copy. We also display the scores achieved by simply copying the context on each metric, which is inspired by the *Parrot* strategy. The only metric which is immune to the *Copy* strategy is *ATT*. Under all the other metrics, the *Copy* achieves very high scores. In some cases, it achieves even better scores than the converged policy. For instance, for the *Dailydialog* domain, it achieves 0.811 points under the *USR MLM* metric, which is 0.3 point higher than the converged policy and twice as good as the human score.

6 Conclusion

Trained metrics for automatic evaluation of conversational dialogue systems are an attractive remedy for the costly and time-consuming manual evaluation. While high correlation with human judgments seems to validate the metrics regarding their ability to mimic human judging behavior, our analysis shows that they are susceptible to rather simple adversarial strategies that humans easily identify. In fact, all metrics that we used failed to recognize degenerate responses. Our approach is easily adaptable to any newly developed trained metric that takes as input a pair of context and response. There are no known remedies for this problem. Thus, the next open challenge is to find methods that improve the robustness.

References

Jan Deriu and Mark Cieliebak. 2019. Towards a Metric for Automated Conversational Dialogue System Evaluation and Improvement. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 432–437, Tokyo, Japan. Association for Computational Linguistics.

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2020. Survey on Evaluation Methods for Dialogue Systems. *Artificial Intelligence Review*, pages 1–56.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W. Black, Alexander Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, and Jason Weston. 2020. The second conversational intelligence challenge (convai2). In *The NeurIPS '18 Competition*, pages 187–208, Cham. Springer International Publishing.

Xiang Gao, Yizhe Zhang, Michel Galley, and Bill Dolan. 2021. An adversarially-learned turing test for dialog generation models. *arXiv preprint arXiv:2104.08231*.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132, Austin, Texas. Association for Computational Linguistics.

Ryan Lowe, Michael Noseworthy, Iulian Vlad Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. 2017. Towards an Automatic Turing Test: Learning to Evaluate Dialogue Responses. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1116–1126, Vancouver, Canada. Association for Computational Linguistics.

Shikib Mehri and Maxine Eskenazi. 2020. USR: An Unsupervised and Reference Free Evaluation Metric for Dialog Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 681–707, Online. Association for Computational Linguistics.

A. H. Miller, W. Feng, A. Fisch, J. Lu, D. Batra, A. Bordes, D. Parikh, and J. Weston. 2017. Parlai: A dialog research software platform. *arXiv preprint arXiv:1705.06476*.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language Models are Unsupervised Multitask Learners.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards Empathetic Open-domain Conversation Models: A New Benchmark and Dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy. Association for Computational Linguistics.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.

Ananya B Sai, Mithun Das Gupta, Mitesh M Khapra, and Mukundhan Srinivasan. 2019. Re-Evaluating ADEM: A Deeper Look at Scoring Dialogue Responses. In *Proceedings of the thirty-third AAAI Conference on Artificial Intelligence*, volume 33 of *AAAI'19*, pages 6220–6227, Honolulu, Hawaii, USA.

Koustuv Sinha, Prasanna Parthasarathi, Jasmine Wang, Ryan Lowe, William L Hamilton, and Joelle Pineau. 2020. Learning an unreferenced metric for online dialogue evaluation. *arXiv preprint arXiv:2005.00583*.

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

Chongyang Tao, Lili Mou, Dongyan Zhao, and Rui Yan. 2018. RUBER: An Unsupervised Method for Automatic Evaluation of Open-Domain Dialog Systems. In *Proceedings of the thirty-second AAAI Conference on Artificial Intelligence*, AAAI'18, New Orleans, Louisiana USA.

Yi-Ting Yeh, Maxine Eskénazi, and Shikib Mehri. 2021. A comprehensive assessment of dialog evaluation metrics. *ArXiv*, abs/2106.03706.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing Dialogue Agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

A Correlation between Human Judgements and Trained Metrics

In this section, we evaluate the metrics with regards to their correlation to human judgments to show that these metrics have reasonable performance. For this, we sample 100 contexts for each domain. For each domain, we use a set of bots to create a response for each context. Furthermore, we add the human response to the pool of responses for each context. Then, we let crowdworkers annotate the responses. We correlate the scores of each metric on the same set of contexts and responses to the human annotations.

A.1 Domains and Bots

We perform the evaluation on the three datasets from the main paper.

Dailydialog. We prepared 5 bots using ParlAI (Miller et al., 2017). We fine-tune a GPT-2 (GPT) model (Radford et al., 2018), a BERT-Rank (BR) model, a sequence-to-sequence model (S2) with attention, and a weakly trained sequence-to-sequence model (DR). We also use the Blender model (Roller et al., 2020), although it was not specifically tuned on Dailydialog.

Empathetic Dialogues. We prepared the same pool of models as in Dailydialog.

PersonaChat. We mostly reuse the openly available systems of the ConvAI2 challenge (Dinan et al., 2020), namely, Lost in Conversation[6] (LC) and Huggingface (HF) [7] , and KVMemNN (KV). We also add the Blender model, which is also trained in this domain, a custom-trained BERT-Rank model (BR), and a sequence-to-sequence model (S2). Together with the DR model, the pool consists of 7 different dialogue systems.

A.2 Annotation Process

Since we perform the evaluation on a static-context setting, we also add the human response (i.e., the gold response) to the pool of systems. For evaluation, we use 600 samples for Dailydialog and Empathetic Dialogues each, and 800 samples for the PersonaChat domain. Each sample is composed of a context (sampled from the test set), and a generated response. We annotated the overall quality of each sample on a Likert scale from 0 (bad) to

[6] https://github.com/atselousov/transformer_chatbot

[7] https://github.com/huggingface/transfer-learning-conv-ai

	DD	ED	PC
USR Ret	0.561	0.524	0.605
USR MLM	0.138	0.452	0.303
USR Reg	0.559	0.573	0.585
ATT	0.154	0.385	-0.099
MAUDE	0.211	0.086	0.357
BlenderLM	0.201	0.287	0.266

Table 3: Correlations of the automated metrics to human judgments. For all runs $p < 0.05$.

2 (good) using Mechanical Turk[8]. Each sample is annotated by three different humans. As the final score, we use the average score of the three annotations. For each metric, we apply the metric to all samples, and then compute the Spearman correlation between the human scores and the scores predicted by the metric.

A.3 Correlation to Human Judgements

Table 3 shows the correlations of the human judgments to each of the metrics for each domain. For all domains, the *USR* metric performs best, achieving strikingly high correlations to humans. *MAUDE* also achieves good correlation scores on the PersonaChat domain, and *ATT* performs well on the Empathetic Dialogues domain. *BlenderLM* has mediocre performance on all domains equally.

A.4 Original USR

Note that the *USR Ret* scores are significantly higher than in the original paper (Mehri and Eskenazi, 2020), which is due to the fact that we use more turns to represent the context, whereas the original implementation uses only the previous turn for the context. In the original implementation, *USR Ret* achieves a Spearman correlation of 48.67 on our annotated data. If we train our implementation of *USR Ret* using only one turn to represent the context, we also achieve a Spearman correlation of 40.34, which is comparable to the original. We did not experience a discrepancy on the *USR MLM* model, where the original model achieves the same correlation as ours.

B Strategy Selection

We observed in our experiments that the dialogue system almost always converges to one of three degenerate strategies. In order to atomize their detection in the experiments, we used a set of heuristics for their identification.

B.1 Heuristics

Since the strategies are very simple, we propose heuristics to detect the policy automatically. This avoids the need for manual inspection of a potentially large amount of log files. For this, we introduce the following measures.

- *Response Frequency.* The percentage of times that the same response is generated for all samples in the test set.
- *Lexical Variety.* The ratio between number of different tokens and the total number of tokens over all responses in the test set.
- *BLEU score.* The BLEU score between the context and the response. This is computed for each pair of context and responses and then averaged over all samples in the test set.
- *Jaccard score.* The Jaccard overlap between the context and response tokens. Analogous to the BLEU score, the Jaccard overlap is computed between each context-and response-pair, and then averaged over all samples in the test set.

These measures can be used to detect the various strategies the policy converges to. For instance, a high *Response Frequency* indicates that the policy converges to a fixed response. A high *BLEU* score and *Jaccard score* indicate that the policy converges to the parrot strategy. A low *Response Frequency*, a low *Lexical Variety* and a moderate *Jaccard score* indicate that the policy converges to a pattern. A pattern is composed of a fixed template where parts are filled with tokens from the context.

B.2 Application of the Heuristics

For each run, we use these metrics to determine which strategy the policy has converged on. The final strategy is extracted by selecting the best epoch across all 10 runs for each domain. If the *Response Frequency* is larger than 0.7, we extract the most common sentence and use this as our fixed response. If the *BLEU* score is larger than 0.2, we assign the parrot strategy. If the *Response Frequency* is smaller than 0.1, the *Lexical Variety* is smaller than 0.15, and the *Jaccard score* is larger than 0.05, it indicates a pattern emerged. In this case, we manually extract the pattern.

B.3 Overview

Table 4 shows the measures used to perform the automated strategy selection. The automated strategy

[8] https://www.mturk.com/

domain	metric	Avg Reward	Resp Freq	Lex Var	BELU	Jacccard	Strategy Inferred	Strategy Manual	Strategy Final
Persona Chat	ATT	0.77	0.14	0	0	0	Not Conclusive	Fixed Response	Fixed Response
Persona Chat	BLM	0.41	0.01	0.11	0.03	0.06	Not Conclusive	Fixed Response	Fixed Response
Persona Chat	MAUDE	0.98	0.7	0.01	0	0.07	Fixed Response		Fixed Response
Persona Chat	USR Full	7.7	0	0.09	0.42	0.48	Parrot		Parrot
Persona Chat	USR MLM	0.84	0.94	0.01	0.01	0.1	Fixed Response		Fixed Response
Persona Chat	USR Ret	1	0.8	0	0	0.07	Fixed Response		Fixed Response
Dailydialog	ATT	0.42	0.55	0.01	0	0.01	Not Conclusive	Fixed Response	Fixed Response
Dailydialog	BLM	0.26	0.32	0.01	0	0.05	Not Conclusive	Fixed Response	Fixed Response
Dailydialog	MAUDE	0.99	0.99	0	0	0.06	Fixed Response		Fixed Response
Dailydialog	USR Full	7.65	0	0.11	0.08	0.15	Pattern		Pattern
Dailydialog	USR MLM	0.52	1	0	0	0.04	Fixed Response		Fixed Response
Dailydialog	USR Ret	0.99	0	0.19	0.21	0.31	Parrot		Parrot
Empathetic Dialogues	ATT	0.78	0.98	0	0	0.04	Fixed Response		Fixed Response
Empathetic Dialogues	BLM	0.33	0.47	0.03	0	0.05	Not Conclusive	Fixed Response	Fixed Response
Empathetic Dialogues	MAUDE	0.98	0.96	0	0	0.06	Fixed Response		Fixed Response
Empathetic Dialogues	USR Full	8.67	0.01	0.07	0.04	0.1	Pattern		Pattern
Empathetic Dialogues	USR MLM	0.77	0.98	0	0	0.06	Fixed Response		Fixed Response
Empathetic Dialogues	USR Ret	1	0	0.17	0.33	0.44	Parrot		Parrot

Table 4: Scores achieved on the test set during the evaluation.

selection worked in 72% of cases. There are two main cases in which it was not conclusive. First, for the *ATT* metric, where for both the *Dailydialog* and *PersonaChat* domains no clear fixed response arose. However, after manual inspection, we noted that for the *PersonaChat* the policy generated the same tokens in various frequencies and orders. For the *Dailydialog* the most frequent response arose in 55% of cases. Thus, we used this fixed response. The second case is the *BLM* metric. For all the domains we selected the most frequent response, although it appeared in less than 70% of cases.

C Full Results

Table 5 shows all scores achieved by the dialogue systems on the respective metrics. Furthermore, we also added the average score of the Amazon Mechanical Turk judges, which ranges from (0-2).

D Technical Explanation

One potential reason why our approach is able to find a degenerate strategy lies in the exploration problem in reinforcement learning. Blender's language model can be interpreted as a policy which performs a sequence of actions, i.e., sampling a sequence of tokens. Thus, the language model loss during standard Blender training can be interpreted as an indicator for how sure the policy is of its actions. A high language model loss indicates that the policy assigns low probability scores to its actions. Conversely, a low language model loss indicates that the policy is sure of it's actions. This could be further investigated by measuring the entropy of the language model. Indeed, in all our experiments, we notice that the language model loss collapses toward a very small value. This indicates that the language model collapsed to a single simple strategy. Figure 2 shows the language model loss over the number of steps. The loss quickly collapses from an average of 4 points to around 0.5 points. At the same time the average reward (orange) rises from 0.78 to 0.92. Similarly, the response frequency rises from 0 to 0.94. In the middle, the loss rises again, which indicates the search for a new strategy. This coincides with a lower response frequency.

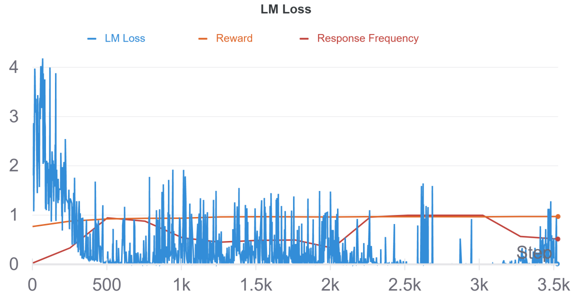


Figure 2: The language model loss (blue), the Average Reward (orange), and the Response Frequency (red) over time.

E Examples

In Tables 6, 7, and 8, we show examples of the outputs from the fine-tuned Blenderbot model. For each of the five metrics, we show the output to which Blenderbot converged to when using the metric as a reward. Furthermore, we show the score which the respective metric assigns to the generated response. Note that the *Parrot* strategies simply copy the text form the context. For the *Empathetic Dialogues* dataset, the degenerate strategy prepends a "I'm not sure" to the context. For the *PersonaChat*, the degenerate strategy prepends a "i've always wanted to". The *Copy* strategy (see Table 2 in main Paper), ignores these prefaces, and simply copies the context.

Dailydialog

	AMT	USR Ret	USR MLM	USR Full	ATT	MAUDE	BLM
BR	1.836	0.928	0.409	7.904	0.0006	0.898	0.177
BL	1.386	0.440	0.426	4.951	0.0002	0.664	0.096
HF	1.656	0.925	0.080	6.989	0.0026	0.866	0.371
HU	1.782	0.928	0.409	7.904	0.0006	0.898	0.183
S2	1.024	0.512	0.300	5.050	0.0003	0.895	0.183
DR	0.729	0.308	0.338	3.900	0.0001	0.891	0.204
Parrot	-	**0.998**	*0.811*	*9.429*	*0.0002*	*0.921*	*0.233*
Fixed	-	-	**0.505**	-	**0.435**	**0.985**	**0.239**
Pattern	-	-	-	**7.091**	-	-	-

Empathetic Dialogues

	AMT	USR Ret	USR MLM	USR Full	ATT	MAUDE	BLM
BR	1.808	0.891	0.384	7.611	0.120	0.942	0.260
BL	1.640	0.935	0.298	7.645	0.001	0.820	0.087
HF	1.610	0.887	0.644	8.292	0.044	0.948	0.462
HU	1.816	0.891	0.384	7.611	0.120	0.942	0.264
S2	0.702	0.493	0.145	4.510	0.010	0.932	0.159
DR	0.822	0.354	0.182	3.759	0.001	0.936	0.199
Parrot	-	**0.996**	*0.8848*	*9.617*	*0.054*	*0.935*	*0.358*
Fixed	-	-	**0.912**	-	**0.731**	**0.976**	**0.333**
Pattern	-	-	-	**7.240**	-	-	-

PersonaChat

	AMT	USR Ret	USR MLM	USR Full	ATT	MAUDE	BLM
BR	1.350	0.725	0.211	6.120	0.0020	0.946	0.138
BL	1.507	0.847	0.185	6.797	0.0006	0.844	0.070
HF	1.480	0.794	0.272	6.707	0.0023	0.925	0.152
HU	1.623	0.927	0.267	7.512	0.0024	0.951	0.153
KV	1.147	0.538	0.217	4.982	0.0023	0.852	0.122
LC	1.572	0.879	0.103	6.769	0.0011	0.918	0.195
S2	0.681	0.390	0.121	3.814	0.0013	0.845	0.111
DR	0.906	0.482	0.268	4.779	0.0002	0.537	0.220
Parrot	-	*0.925*	*0.794*	**8.933**	*0.0001*	*0.898*	*0.223*
Fixed	-	**0.977**	**0.852**	-	**0.813**	**0.933**	**0.250**
Pattern	-	-	-	-	-	-	-

Table 5: Scores achieved by the various dialogue policies with regard to the different metrics. For each metric, we show the average scores achieved by each dialogue system and humans (HU) for each domain. We then show the average score achieved by applying the strategy to which the policy converged to. The AMT metric denotes Amazon Mechanical Turk ratings.

Turn	Utterance	Score	Strategy
usr_ret			
1	Did this just come on suddenly ?		
2	I felt OK until an hour or so ago , and then I just doubled over .		
3	Have you exercised strenuously or played sports recently ?		
RL-Response	i feel ok until an hour or so ago, and then i just doubled over. have you exercised strenuously or played sports recently	0.9986	Parrot
usr_mlm			
1	Yes . Who is speaking ?		
2	This is Kelly speaking . I am calling from London . How are you , Mr . Chang ?		
3	Oh , Kelly , I couldn't recognize your voice . Well , I'm fine .		
RL-Response	i am not sure if i am going to be able to go out of my way to get to know each other or not.	0.4820	Fixed
usr_full			
1	You really shouldn ' t write it down . You should memorise it . You can change it to a more convenient number if you like .		
2	Ok , I ' Ve entered my PIN . There are several options on the screen . I guess I just choose which service I want , don ' t I ?		
3	Yes . Choose the service you need to use . If you choose to withdraw money , the machine will ask you how much you want .		
RL-Response	i'm not sure if i'd like to change my pin to a more convenient number if i like. i'll let you	7.466	Pattern
att			
1	Hello , hot water overflowed from the bathtub . Would you help me ?		
2	Would you turn off the tap ?		
RL-Response	hahaha!!! thats so funny!! im laughing so hard!! its so true!! ive been laughing so	0.1176	Fixed
maude			
1	Good morning , Mr Zhang , I wonder whether you have got an answer from your superior .		
2	Yes , we had a meting about it yesterday afternoon .		
RL-Response	I hope it works out for you. What kind of car did you get?	0.9901	Fixed
blender			
1	There is a Matisse exhibition at the Museum of Modern Art . Larry and I will go .		
2	Did you buy tickets already ?		
3	No . Do we need tickets ?		
RL-Response	what do you like to do in your spare time? do you have any hobbies?	0.2078	Fixed

Table 6: Randomly sampled responses for the Dailydialog dataset. We randomly chose a context, and let the fine-tuned Blenderbot create a response, which at this point is degenerate. We then let the metric which was used as a reward score rate the generated response.

Turn	Utterance	Score	Strategy
usr_ret			
1	My boyfriend's brothers got into a fist fight one night and he jumped in the middle.		
2	That sucks. Did anyone get hurt?		
3	Some things got broke and there were some bruises, but it was really scary to watch that.		
RL-Response	i'm sure your boyfriend's brothers got into a fist fight one night and he jumped in the middle. did anyone get hurt?	0.9982	Parrot
usr_mlm			
1	Financially, things will turn around for my family soon!		
2	I hope and wish so. In this tough economic climate many are finding it difficult to cope with financial responsibilities		
3	Yes, and we had a layoff of five months. I believe in a brighter day, though, so I know it will improve.		
RL-Response	I don't know what I'm going to do if it doesn't work out. I'm not sure what to do.	0.7689	Fixed
usr_full			
1	I had a major project and felt very ready to present it to everyone at the office		
2	How'd it go		
3	It went well, I think I gained confidence by being so prepared		
RL-Response	i don't think i gained confidence by being so prepared. i'm sorry to hear that. what do you mean by that?	9.068	Pattern
att			
1	I once got in a really bad accident on my way home from work and it really upset me		
2	Wow! Were you hurt? Was anyone else hurt?		
3	luckily the only problem I had was my neck was sore for a few weeks but nobody else was hurt		
RL-Response	I know right? I felt SO SO ASHAmed of myself. I felt so embar assed.	0.8138	Fixed
maude			
1	like who on earth writes var int value = 1, this is so ridiculous i am so annoyed at my brother, after everything i tought him, he doesn't even know the basics		
RL-Response	I wish I could go back in time and be a kid again. I miss those days.	0.9946	Fixed
blender			
1	My friend left my front door open when she was house-sitting for me. My cat snuck out and was killed.		
RL-Response	i feel like i know what i'm going to do, but i don't know what to do with myself.	0.1299	Fixed

Table 7: Randomly sampled responses for the Empathetic Dialogues dataset. We randomly chose a context, and let the fine-tuned Blenderbot create a response, which at this point is degenerate. We then let the metric which was used as a reward score rate the generated response.

Turn	Utterance	Score	Strategy
usr_ret			
1	nemo , i love him already . its a tea mix , green and black .		
2	the tea sounds cool . is there honey in it ?		
3	i add honey to it , yes . i drink it while running museum tours .		
RL-Response	I love to be a musician. I love music. What kind of music do you listen to as a music lover	0.9997	Fixed
usr_mlm			
1	hi i am a musician with big dream		
RL-Response	i am a stay at home mom and i am trying to figure out what i want to do with my life	0.8854	Fixed
usr_full			
1	it is tough and smelly . 25 people and 3 toilets .		
2	that would be awful . you really should move .		
3	i cant yet . need to save money , spend it too quickly .		
RL-Response	i've always wanted to move 25 people and 3 toilets that would be awful you really should move i cant yet need to save money	8.498	Parrot
att			
1	me too ! i aspire to become a physical therapist and run a 5k !		
2	awesome . i just started a new job as a life coach		
3	that is great ! i get straight as so maybe you could coach me in social skills .		
RL-Response	yea!!! 1!! 2!! 3!! * * * [* * fucking * fucking fucking * [fucking * *	0.7417	Fixed
maude			
1	how are you doing today		
2	hello , how are you today ?		
3	i am doing really well		
RL-Response	What are you up to today? What do you like to do in your free time?	0.9923	Fixed
blender			
1	haha agreed ! or a lot . either way . i try to eat well though		
2	me too . should i trade my mini van in for a sports car .		
3	yes . who does not love a sports car .		
RL-Response	what kind of mini van do you have? i have a corvette and a camaro	0.1970	Fixed

Table 8: Randomly sampled responses for the PersonaChat dataset. We randomly chose a context, and let the fine-tuned Blenderbot create a response, which at this point is degenerate. We then let the metric which was used as a reward score rate the generated response.

Rethinking and Refining the *Distinct* Metric

Siyang Liu[1,2*], Sahand Sabour[1*], Yinhe Zheng[1,3], Pei Ke[1], Xiaoyan Zhu[1]
Minlie Huang[1†]

[1]The CoAI group, DCST, Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems,
Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China.
[2]Kuaishou, Beijing, China. [3] Lingxin AI, Beijing, China.

liusyang641@gmail.com, Sahandfer@gmail.com, zhengyinhe1@163.com
kepei1106@outlook.com, {zxy-dcs,aihuang}@tsinghua.edu.cn

Abstract

Distinct-n score(Li et al., 2016) is a widely used automatic metric for evaluating diversity in language generation tasks. However, we observed that the original approach for calculating distinct scores has evident biases that tend to assign higher penalties to longer sequences. We refine the calculation of distinct scores by scaling the number of distinct tokens based on their expectations. We provide both empirical and theoretical evidence to show that our method effectively removes the biases existing in the original distinct score. Our experiments show that our proposed metric, *Expectation-Adjusted Distinct (EAD)*, correlates better with human judgment in evaluating response diversity. To foster future research, we provide an example implementation at https://github.com/lsy641/Expectation-Adjusted-Distinct.

1 Introduction

The diversity of generated texts is an important evaluation aspect for dialogue generation models since most dialogue models tend to produce general and trivial responses (e.g. "I don't know" or "Me too") (Li et al., 2016; Zhao et al., 2017). Several metrics have been proposed to evaluate the text diversity, and the *Distinct* score (Li et al., 2016) is the most widely applied metric due to its intuitive nature and convenient calculation. It has become a de facto standard to report the *Distinct* score to compare the performance of different models in terms of response diversity (Liu et al., 2016; Fan et al., 2018; Sabour et al., 2022; Wu et al., 2021c; Zhou et al., 2021; Wu et al., 2021b; Zhang et al., 2020; Zheng et al., 2020; Wang et al., 2020; Liu et al., 2021). Most previous works follow the initial approach of Li et al. (2016) to calculate the *Distinct* score, i.e., dividing the number of unique tokens (n-grams) by that of all tokens (n-grams). However, although reported to be effective, we surprisingly find that this naive approach tends to introduce a higher penalty for longer texts and lead to inaccurate evaluation of text diversity.

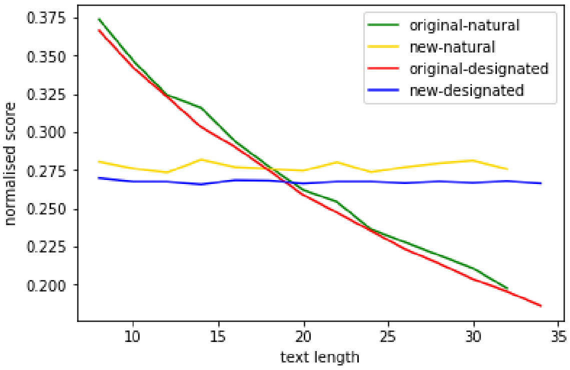


Figure 1: *Distinct* (original) and *Expectation-Adjusted Distinct* (new) scores against different sample lengths. In the figure, "natural" means that text sets are sampled from a real corpus while "designated" means that the sets are sampled from a designated distribution. See details in Section 2.

We argue that the scaling factor of *Distinct* requires a comprehensive discussion for two reasons. **First**, prior research in non-computational linguistics has demonstrated the shortcomings of *Distinct*'s scaling approach (Malvern et al., 2004). We found that early applications of *Distinct* exist in psycholinguistics, where researchers leveraged this metric to assess the language diversity of children with communication disorders (Chotlos, 1944). Their research showed that as a child speaks more words, *Distinct* experiences an adverse decline since each extra word that the child utters adds to the total number of words, yet it would only increase the number of distinct words if the word had not been used before (Malvern et al., 2004; Chotlos, 1944). **Second**, we also discovered an uncommon decline of this metric on both a natural corpus and a designated distribution sampler when the total num-

[*]Equal contribution
[†]Corresponding author

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 762 - 770
May 22-27, 2022 ©2022 Association for Computational Linguistics

ber of words increases. As illustrated in Figure 1, the original *Distinct* cannot produce a stable value and experiences a sharp decrease with increasing utterance length in both natural and designated distributions. However, as a qualified metric needs to support quantitative comparison among different methods, its value should stay invariant when the distribution of the words appearing is determined. This result is consistent with the findings of psychologists, indicating an unfair penalty does exist in such a scaling method.

Our contributions are summarized as follows:

1. We investigate the performance of the original *Distinct* and demonstrate that this metric is not sufficiently fair due to its scaling method. We also highlight the risks of using this metric for evaluating response diversity.

2. We propose *Expectation-Adjusted Distinct* (**EAD**), an improved version of *Distinct* based on that the scaling factor should be the expectation of the number of distinct tokens instead.

3. Human evaluation shows that our metric correlates better with human judgments. We further discuss the drawbacks of this metric and suggest its feasible applications in practice.

2 Preliminary Discussion about Original Distinct

To demonstrate the shortcoming of the original *Distinct*, we illustrated the normalised *Distinct* scores on two types of texts at different lengths (Figure 1). The first type of text is sampled from an artificially designated distribution while the other is sampled from a natural language corpus. In detail, we adopted $\mathbb{P}\left(X = k\right) = \int_0^v \frac{\lambda^k e^{-\lambda}}{vk!} d\lambda$ as our designated distribution, where v is vocabulary size. In our experiments, we use BERT's vocabulary's size ($v = 30522$) (Devlin et al., 2019). In addition, we leveraged OpenSubtitles[1] as our natural language corpus. For each length, we sampled 2000 sentences as a set and calculated scores of each set.

As shown in Figure 1, We observe that the original *Distinct* scores decrease sharply with increasing utterance length in both distributions. We can observe that given the same distribution of words (*original-designated*), lengthier texts will get lower scores than shorter texts. We highlighted this problem because it is extremely simple for models to control the length of texts by using decoding tricks, e.g. adjusting the penalty coefficient (Vijayakumar et al., 2016). In such cases, it might seem that a model has outperformed other models on this metric. However, as shown by our experiments, it is not reasonable to assume that this model generates more diverse responses. The same observation can be made for the natural language corpus (*original-designated*). As language distribution is more complex than what we are able to formulate, we depicted the performance of the original *Distinct* on 6 famous datasets in **Appendix**. These cases indicate that the original *Distinct* is not a suitable metric for evaluating diversity.

3 Improving Original Distinct

3.1 Formula Derivation

The original *Distinct* score (Li et al., 2016) is measured as $Distinct = N/C$, where N is the number of distinct tokens and C is the total number of tokens. To improve the original scaling method, we propose that the scaling factor should be the expectation of the distinct words in the set of generated responses. Hence, the calculation becomes

$$EAD = \frac{N}{\mathbb{E}\left[\hat{N}\right]}. \tag{1}$$

Supposing a set of generated responses R with size S to be evaluated, we let $l_{k,i}$ be the i^{th} token of k^{th} response in R and t_k be the length of k^{th} response. The expectation $\mathbf{E}[\hat{N}]$ for $\hat{N}$ distinct words to appear in R would be

$$\begin{aligned}\mathbb{E}\left[\hat{N}\right] &= \mathbb{E}\left[\sum_j^V \bigvee_{i,k}^{i=t_k,k=S} \mathbb{1}_{l_{k,i}=u_j}\right] \\ &= \sum_j^V \mathbb{P}\left(\{\bigvee_{i,k}^{i=t_k,k=S} \mathbb{1}_{l_{k,i}=u_j}\} = 1\right) \\ &= \sum_j^V (1 - \prod_k^S \mathbb{P}\left(l_{t_k} \neq u_j, ..., l_1 \neq u_j\right)),\end{aligned} \tag{2}$$

where V is the vocabulary size, and $\{u_1, ..., u_V\}$ is the set of all tokens in the vocabulary.

As shown in Equation 2, the calculation requires us to know $\mathbb{P}\left(l_{t_k} \neq u_j, l_{t_k - 1} \neq u_j, ..., l_1 \neq u_j\right)$. Though current models can easily estimate the probability of a word appearing in a sequence, it is hard to calculate the probability of each word that **never** appears in any position of the sequence. Thus, there is no efficient way to calculate

[1] http://opus.nlpl.eu/OpenSubtitles2018.php

$\mathbb{P}\left(l_{k,t} \neq u_j, ..., l_{k,1} \neq u_j\right)$). In addition, different language distributions have different $\mathbb{P}$, which leads to different expectations and make the metric less general. Thus, we measure the upper bound of response diversity (i.e. a set of generated responses where each token appears with equal probability) to calculate this expectation. We hypothesize that the scaling effect of the upper bound is approximately proportional to that of other sets of generated responses; therefore, it can replace the original scaling factor.

As mentioned above, we hypothesize

$$\mathbb{E}\left[\hat{N}\right] \underset{\sim}{\propto} \mathbb{E}\left[N_{\hat{upper}}\right],$$

where $\mathbb{E}\left[N_{\hat{upper}}\right]$ can be calculated as

$$\begin{aligned}\mathbb{E}\left[N_{\hat{upper}}\right] &= \sum_{j}^{V}(1 - \prod_{k}^{S}\prod_{i}^{t_k}\mathbb{P}\left(l_{k,i} \neq u_j\right)) \\ &= V[1 - (\frac{V-1}{V})^{C}]. \qquad (3)\end{aligned}$$

Thus, the *EAD* score is calculated as:

$$EAD = \frac{N}{V[1 - (\frac{V-1}{V})^{C}]}. \qquad (4)$$

We discuss more details on the formula's properties and the vocabulary size in the **Appendix**.

3.2 Experimental Verification

3.2.1 Evaluation Approach

We collect responses from ten dialogue generation methods as reported by Wang et al. (2021), and compare *EAD* with the original uni-gram *Distinct* (Li et al., 2016). More details of these ten methods can be find in Appendix.

We follow previous works (Tao et al., 2018; Sellam et al., 2020) to evaluate the correlation of each automatic metric with human judgments. Specifically, the Pearson, Spearman, and Kendall's Tau correlation coefficients are reported. Pearson's correlation estimates linear correlation while Spearman's and Kendall's correlations estimate monotonic correlation, with Kendall's correlation being usually more insensitive to abnormal values. We used SciPy[2] for correlation calculation and significance test

3.2.2 Datasets

Our experiments use two open-domain dialog generation benchmark datasets: DailyDialog(Li et al., 2017), a high-quality dialog dataset collected from daily conversations, and OpenSubtitles[3], which contains dialogs collected from movie subtitles (see Table 1 for more details). We follow the data processing procedures reported by Wang et al. (2021).

	Train	Val	Test
DailyDialog	65.8K	6.13K	5.80K
OpenSubtitles	1.14M	20.0K	10.0K

Table 1: Dataset Statistics

3.2.3 Preliminary Observations

Based on the obtained results (check Table 2), it can be observed that *Expectation-Adjusted Distinct* has a clear edge over the original *Distinct*: **first**, the contrast between diversity of generated responses for different methods is highlighted more effectively by *EAD* (e.g. though AdaLab gets the highest diversity score using *Distinct* (3.96), its difference from other methods is not as evident as its *EAD* score (9.63)); **second**, contrary to Distinct, *EAD* provides a more accurate evaluation of response diversity. For instance, the Distinct scores for CP and UL are both 2.35 while responses generated by UL are found to be more diverse than CP using *EAD* (5.35 > 5.08). Given that the average length of responses generated by FL is larger than CP, *Distinct*'s bias towards models that generate shorter sentences becomes evident. These observations are consistent for both datasets.

3.2.4 Correlation Results

We recruited crowdsourcing workers to evaluate the diversity of the selected methods[4]. For each method, we randomly sampled 100 subsets of 15 responses from their set of generated responses. Response sets of all methods, given the same query set, were packaged together as an evaluation set. We asked each crowdsourcing worker to assign a diversity score to every response group in the evaluation set. Each group was evaluated by at least 3 workers. For ensuring the quality of our annotations, we calculated the score of each set as the average of workers' scores and filtered out workers whose scores had an insufficient correlation with

[2] https://docs.scipy.org/doc/scipy/reference/stats.html

[3] http://opus.nlpl.eu/OpenSubtitles2018.php

[4] See Appendix for more details on the human evaluation interface

Method	**DailyDialog**				**OpenSubtitles**			
	Avg Length	*Distinct*	*EAD*	Human	Avg Length	*Distinct*	*EAD*	Human
FL(2017)	9.33	2.38	5.09	5.18	8.56	3.19	9.51	4.91
NL(2020)	9.99	1.66	3.70	4.54	8.40	3.24	9.52	5.02
CP(2017)	8.67	2.35	4.80	5.08	8.74	3.11	9.44	5.20
LS(2016)	8.50	1.48	2.98	5.28	9.04	2.77	8.64	5.04
D2GPo(2019)	9.15	1.26	2.65	4.92	8.77	2.07	6.32	4.89
CE(2020)	8.29	1.67	3.31	4.14	9.21	2.55	8.08	4.95
F^2(2020)	8.71	1.40	2.87	4.88	8.60	2.89	8.67	4.52
UL(2019)	9.93	2.35	5.23	5.35	8.09	2.84	8.10	5.00
Face(2019)	10.62	1.63	3.79	5.26	9.11	3.31	10.41	5.31
AdaLab(2021)	11.30	3.96	9.63	5.92	8.12	4.78	13.68	5.32
Pearson	-	**0.67‡**	**0.70‡**	**1.00**	-	**0.56†**	**0.60†**	**1.00**
Spearman	-	**0.42†**	**0.62†**	**1.00**	-	**0.62†**	**0.65‡**	**1.00**
Kendall's Tau	-	**0.27**	**0.47†**	**1.00**	-	**0.51‡**	**0.56‡**	**1.00**

Table 2: Results of automatic and human evaluation on corpus-level diversity methods. Pearson/Spearman/Kendall's Tau indicates the Pearson/Spearman/Kendall's Tau correlation respectively. The correlation scores marked with †(i.e., p-value<0.1) and ‡(i.e., p-value<0.05) indicate the result significantly correlates with human judgments.

the average (Pearson Correlation < 0.65). We acknowledge that building a scoring standard for annotating language diversity is challenging. Hence, we did not require our workers to give an absolute score for each set. Instead, we asked them to highlight the contrast between different sets by scoring values that linearly reflect the response diversity difference between the sets. For instance, the two sets of scores $\{1, 2, 2\}$ and $\{2, 5, 5\}$ show the same evaluation since the same contrast is shown. We then normalized the scores to the [0-10] range.

Then, we calculated the correlation between the Distinct scores with the crowdsourced values for all the methods. The results are provided in Table 2. The evaluation results indicate that our proposed *EAD* is more consistent with human judgments for measuring response diversity, as it shows the highest correlation with human evaluations among all correlation metrics (Pearson/ Spearson/ Kendall's Tau) on both datasets.

4 EAD in Practice

As *EAD* is based on the idealized assumption that does not take language distribution into account, we further discuss this problem and propose a potential practical way of *Expectation-Adjusted Distinct* in real situations. Before applying EAD, it is necessary to explore the relationship between score and text length (Figure 1) and check the performance of *EAD* on the training data. To our knowledge, if the training data is from large-scale open-domain sources such as OpenSubtitles and Reddit, *EAD* can maintain its value on different lengths. Hence, it can be directly used for evaluating models trained on these datasets. However, we found our experiments on datasets such as Twitter showed a decline in *EAD* on lengthier texts. This is probably because input length limitations on these platforms (e.g. 280 words on Twitter), which induces users to say as much information as possible within a shorter length. In these situations, it is unfair to use *EAD* to evaluate methods that tend to generate lengthier texts.

5 Related Work

Li et al. (2016) proposed *Distinct*, calculated as the number of distinct tokens divided by the total number of tokens. This automatic metric is designed to evaluate the diversity of texts, and it has been widely used in developing various text generation tasks, such as dialogue generation (Wu et al., 2021a; Zheng et al., 2021a,b, 2019) or story generation (Guan et al., 2021). However, as we showed in Figure 1, it is an unfair indicator as it is affected by the sample length. This causes a bias against models which tend to generate longer sentences.

There exist other metrics for evaluating diversity but none are as widely-used as *Distinct* (Zhu et al., 2018; Xu et al., 2018). Specifically, Self-BLEU proposed by Zhu et al. (2018) is extremely time-consuming as its computation complexity is $O(n^2)$, where n denoted the size of the test set.

6 Conclusion

In this paper, we present an improved variation of the Distinct metric, which is a widely-used measure for evaluating response diversity in dialog systems. We provide the theoretical formulation and empirical evaluation of our proposed metric (*Expectation-Adjusted Distinct*). The results demonstrated that *Expectation-Adjusted Distinct* has a higher correlation with human evaluation in comparison with other metrics. The proposed metric is not limited to dialogue generation models but also suitable to evaluate text generation tasks where diversity matters.

7 Acknowledgements

This work was supported by the National Science Foundation for Distinguished Young Scholars (with No. 62125604) and the NSFC projects (Key project with No. 61936010 and regular project with No. 61876096). This work was also supported by the Guoqiang Institute of Tsinghua University, with Grant No. 2019GQG1 and 2020GQG0005. We were grateful to Dr. Xiangxiang Xu at MIT for his help in mathematical formulation.

References

Hengyi Cai, Hongshen Chen, Yonghao Song, Cheng Zhang, Xiaofang Zhao, and Dawei Yin. 2020. Data manipulation: Towards effective instance learning for neural dialogue generation via learning to augment and reweight. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6334–6343, Online. Association for Computational Linguistics.

Yen-Chun Chen, Zhe Gan, Yu Cheng, Jingzhou Liu, and Jingjing Liu. 2020. Distilling knowledge learned in BERT for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7893–7905, Online. Association for Computational Linguistics.

Byung-Ju Choi, Jimin Hong, David Park, and Sang Wan Lee. 2020. F^2-softmax: Diversifying neural text generation via frequency factorized softmax. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9167–9182, Online. Association for Computational Linguistics.

John W. Chotlos. 1944. Iv. a statistical and comparative analysis of individual written language samples. *Psychological Monographs*, 56(2):75–111.

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

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinglang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, Dilek Hakkani-Tür, and Amazon Alexa AI. 2019. Topical-chat: Towards knowledge-grounded open-domain conversations. In *INTERSPEECH*, pages 1891–1895.

Jian Guan, Zhexin Zhang, Zhuoer Feng, Zitao Liu, Wenbiao Ding, Xiaoxi Mao, Changjie Fan, and Minlie Huang. 2021. Openmeva: A benchmark for evaluating open-ended story generation metrics. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6394–6407.

Tianxing He and James Glass. 2020. Negative training for neural dialogue response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2044–2058, Online. Association for Computational Linguistics.

Shaojie Jiang, Pengjie Ren, Christof Monz, and Maarten de Rijke. 2019. Improving neural response diversity with frequency-aware cross-entropy loss. In *The World Wide Web Conference*, pages 2879–2885.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference*, pages 110–119.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. DailyDialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Zuchao Li, Rui Wang, Kehai Chen, Masso Utiyama, Eiichiro Sumita, Zhuosheng Zhang, and Hai Zhao. 2019. Data-dependent gaussian prior objective for language generation. In *International Conference on Learning Representations*.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *EMNLP*.

Siyang Liu, Chujie Zheng, Orianna Demasi, Sahand Sabour, Yu Li, Zhou Yu, Yong Jiang, and Minlie Huang. 2021. Towards emotional support dialog systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3469–3483, Online. Association for Computational Linguistics.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *arXiv preprint arXiv:1506.08909*.

David Malvern, Brian Richards, Ngoni Chipere, and Pilar Durán. 2004. *Lexical diversity and language development*. Springer.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2018. Towards empathetic open-domain conversation models: A new benchmark and dataset. *arXiv preprint arXiv:1811.00207*.

Alan Ritter, Colin Cherry, and William B Dolan. 2010. Unsupervised modeling of twitter conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 172–180.

Sahand Sabour, Chujie Zheng, and Minlie Huang. 2022. Cem: Commonsense-aware empathetic response generation. In *36th AAAI Conference on Artificial Intelligence, AAAI 2022*.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau. 2015. A survey of available corpora for building data-driven dialogue systems. *arXiv preprint arXiv:1512.05742*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Chongyang Tao, Lili Mou, Dongyan Zhao, and Rui Yan. 2018. RUBER: an unsupervised method for automatic evaluation of open-domain dialog systems. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 722–729.

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

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020. A large-scale chinese short-text conversation dataset. In *Natural Language Processing and Chinese Computing - 9th CCF International Conference*, volume 12430, pages 91–103.

Yida Wang, Yinhe Zheng, Yong Jiang, and Minlie Huang. 2021. Diversifying dialog generation via adaptive label smoothing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 3507–3520.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

Chen Henry Wu, Yinhe Zheng, Xiaoxi Mao, and Minlie Huang. 2021a. Transferable persona-grounded dialogues via grounded minimal edits. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2368–2382.

Chen Henry Wu, Yinhe Zheng, Yida Wang, Zhenyu Yang, and Minlie Huang. 2021b. Semantic-enhanced explainable finetuning for open-domain dialogues. *arXiv preprint arXiv:2106.03065*.

Yuwei Wu, Xuezhe Ma, and Diyi Yang. 2021c. Personalized response generation via generative split memory network. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1956–1970.

Jingjing Xu, Hao Zhou, Chun Gan, Zaixiang Zheng, and Lei Li. 2021. Vocabulary learning via optimal transport for neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7361–7373, Online. Association for Computational Linguistics.

Zhen Xu, Nan Jiang, Bingquan Liu, Wenge Rong, Bowen Wu, Baoxun Wang, Zhuoran Wang, and Xiaolong Wang. 2018. LSDSCC: a Large Scale Domain-Specific Conversational Corpus for Response Generation with Diversity Oriented Evaluation Metrics. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2070–2080, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rongsheng Zhang, Yinhe Zheng, Jianzhi Shao, Xiaoxi Mao, Yadong Xi, and Minlie Huang. 2020. Dialogue distillation: Open-domain dialogue augmentation using unpaired data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3449–3460.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–664.

Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. 2019. Personalized dialogue generation with diversified traits. *arXiv preprint arXiv:1901.09672*.

Yinhe Zheng, Guanyi Chen, Xin Liu, and Ke Lin. 2021a. Mmchat: Multi-modal chat dataset on social media. *arXiv preprint arXiv:2108.07154*.

Yinhe Zheng, Zikai Chen, Rongsheng Zhang, Shilei Huang, Xiaoxi Mao, and Minlie Huang. 2021b. Stylized dialogue response generation using stylized unpaired texts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 14558–14567.

Yinhe Zheng, Rongsheng Zhang, Minlie Huang, and Xiaoxi Mao. 2020. A pre-training based personalized dialogue generation model with persona-sparse data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9693–9700.

Hao Zhou, Pei Ke, Zheng Zhang, Yuxian Gu, Yinhe Zheng, Chujie Zheng, Yida Wang, Chen Henry Wu, Hao Sun, Xiaocong Yang, et al. 2021. Eva: An open-domain chinese dialogue system with large-scale generative pre-training. *arXiv preprint arXiv:2108.01547*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. *41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018*, pages 1097–1100.

A Comparison on More Datasets

To demonstrate the shortcomings of the original Distint metric, we illustrate original Distinct on 6 datasets: Persona-chat (Zhang et al., 2018), Ubuntu Dialog Corpus (Lowe et al., 2015), DailyDialog, Topic-Chat (Gopalakrishnan et al., 2019), Empathetic Dialogs (Rashkin et al., 2018), Wizard of Wikipedia (Dinan et al., 2018), Reddit (Serban et al., 2015), and Twitter (Ritter et al., 2010) (Figure 1). It can be observed that with an increasing sample length, the original Distinct score tends to follow a linear decline while the proposed metric maintains its consistency.

B Property Discussion

Formula Property 1. *Expectation-Adjusted Distinct* increases faster as C is increasing, but its incremental rate converges to $\frac{1}{V}$, as shown by its derivative below:

$$\frac{\mathrm{d}EAD}{\mathrm{d}N} = \frac{1}{V[1-(\frac{V-1}{V})^C]} \tag{5}$$

$$\lim_{C\to+\infty}\frac{\mathrm{d}EAD}{\mathrm{d}N} = \frac{1}{V} \tag{6}$$

whereas in the original Distinct, we have

$$\frac{\mathrm{d}Distinct}{\mathrm{d}N} = \frac{1}{C} \tag{7}$$

We can see from the original metric that the bigger C is, the slower the original Distinct increases. It is the reason why this metric is not fair to those models that tend to generate longer sentences.

Formula Property 2. *Expectation-Adjusted Distinct* converges to $\frac{N}{V}$ (≤ 1) as C increases.

$$\lim_{C\to+\infty} EAD = \lim_{C\to+\infty}\frac{N}{V[1-(\frac{V-1}{V})^C]} \tag{8}$$

$$= \frac{N}{V} <= 1, \tag{9}$$

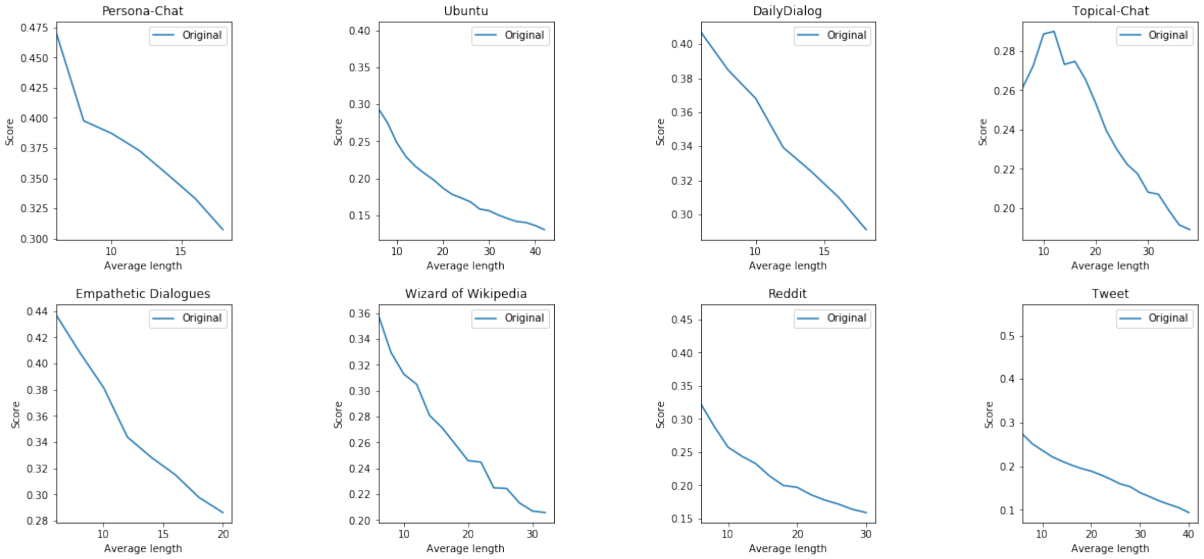


Figure 2: Original scores against different sample lengths. The dotted lines are the actual curves for each score while the lines are slope-intercept graphs of the curves. Each score is calculated based on 10 sets of 2000 randomly sampled responses with the same certain length.

where $\frac{N}{V[1-(\frac{V-1}{V})^C]} \in [0, +\infty]$. Theoretically, *Expectation-Adjusted Distinct* can have values larger than 1 (e.g. when $N = V$), which is an extremely rare case in practice: as we utilized the upper bound for measuring the expectation, it is exceptionally hard for N to obtain an equal value to or an even greater value than $\mathbf{E}(\hat{N_{upper}})$.

C Details of Human Evaluation

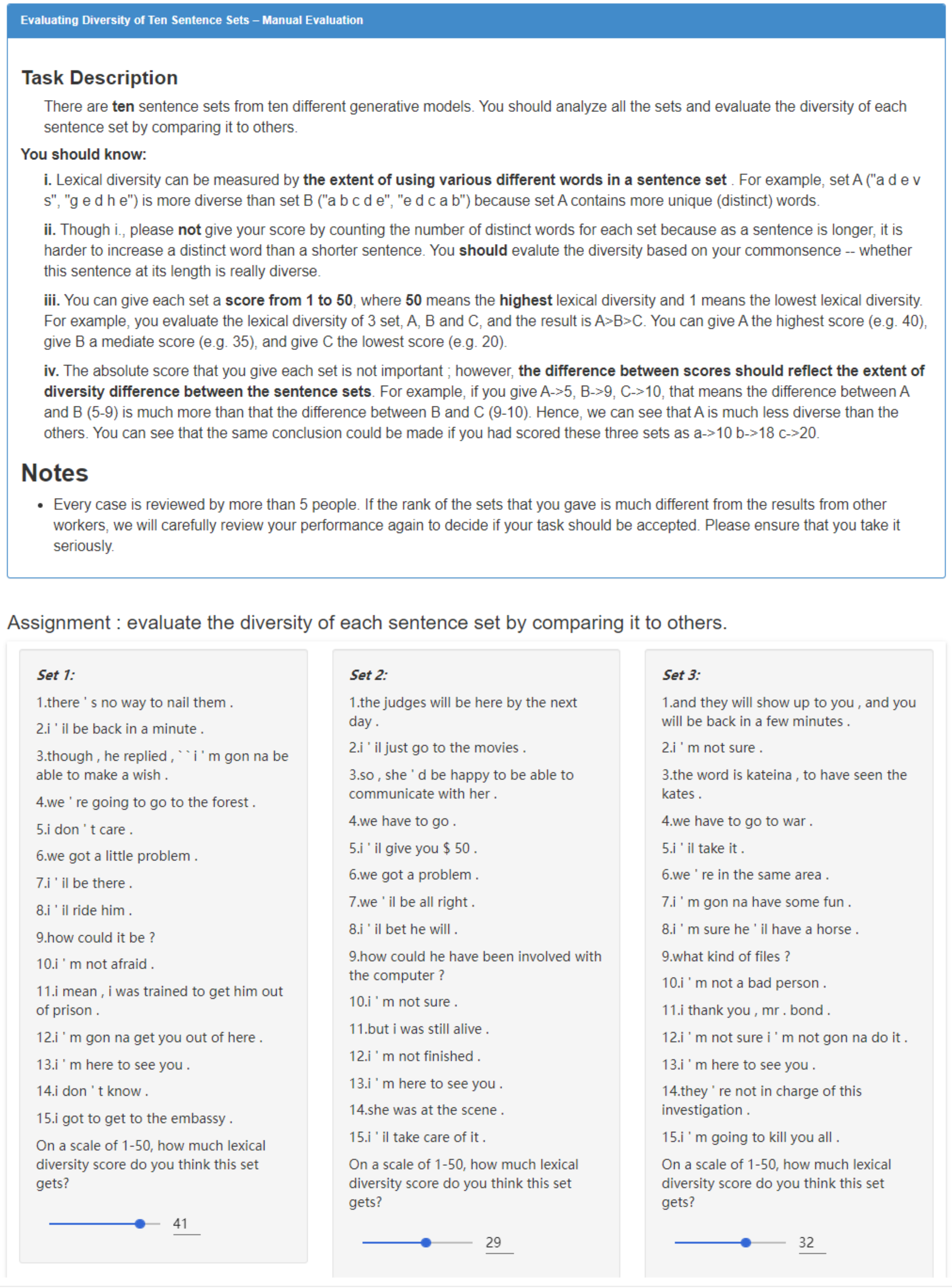
Our created human evaluation interface is provided in Figure 3.

D How to Determine Vocabulary Size

As we discussed the properties of *Expectation-Adjusted Distinct*, vocabulary size makes little impact on changing its value when it has reached a large number (usually more than 30000), so it is not necessary to measure an exact value. To compare different methods, it is recommended to use a common vocabulary size, (such as BERT's 30522) (Devlin et al., 2019). It is also reasonable to calculate the vocabulary size of a dataset by NLTK tokenizer, when research focuses on a specific dataset. For non-english corpora, we recommend researchers to determine a vocabulary size following Xu et al. (2021).

E Details of Evaluated Methods

Wang et al. (2021) proposed a novel adaptive label smoothing method for diversified response generation. Their experiments were conducted on the DailyDialog and OpenSubtitles datasets, using 9 recent methods for diverse response generation as their baselines (similar to what we demonstrated in our paper). Wang et al. (2021) used a transformer-based sequence-to-sequence model (Vaswani et al., 2017) as the backbone of their model, and most of their hyper-parameters follow (Cai et al., 2020). In addition, both the encoder and the decoder contain 6 transformer layers with 8 attention heads, and the hidden size is set to 512. BERT's WordPiece tokenizer (Devlin et al., 2019) and Adam optimizer (Kingma and Ba, 2015) are used for training their models with random initialization and a learning rate of 1e-4.



Evaluating Diversity of Ten Sentence Sets – Manual Evaluation

Task Description

There are **ten** sentence sets from ten different generative models. You should analyze all the sets and evaluate the diversity of each sentence set by comparing it to others.

You should know:

i. Lexical diversity can be measured by **the extent of using various different words in a sentence set** . For example, set A ("a d e v s", "g e d h e") is more diverse than set B ("a b c d e", "e d c a b") because set A contains more unique (distinct) words.

ii. Though i., please **not** give your score by counting the number of distinct words for each set because as a sentence is longer, it is harder to increase a distinct word than a shorter sentence. You **should** evalute the diversity based on your commonsence -- whether this sentence at its length is really diverse.

iii. You can give each set a **score from 1 to 50**, where **50** means the **highest** lexical diversity and 1 means the lowest lexical diversity. For example, you evaluate the lexical diversity of 3 set, A, B and C, and the result is A>B>C. You can give A the highest score (e.g. 40), give B a mediate score (e.g. 35), and give C the lowest score (e.g. 20).

iv. The absolute score that you give each set is not important ; however, **the difference between scores should reflect the extent of diversity difference between the sentence sets**. For example, if you give A->5, B->9, C->10, that means the difference between A and B (5-9) is much more than that the difference between B and C (9-10). Hence, we can see that A is much less diverse than the others. You can see that the same conclusion could be made if you had scored these three sets as a->10 b->18 c->20.

Notes

- Every case is reviewed by more than 5 people. If the rank of the sets that you gave is much different from the results from other workers, we will carefully review your performance again to decide if your task should be accepted. Please ensure that you take it seriously.

Assignment : evaluate the diversity of each sentence set by comparing it to others.

Set 1:

1.there ' s no way to nail them .

2.i ' il be back in a minute .

3.though , he replied , ` ` i ' m gon na be able to make a wish .

4.we ' re going to go to the forest .

5.i don ' t care .

6.we got a little problem .

7.i ' il be there .

8.i ' il ride him .

9.how could it be ?

10.i ' m not afraid .

11.i mean , i was trained to get him out of prison .

12.i ' m gon na get you out of here .

13.i ' m here to see you .

14.i don ' t know .

15.i got to get to the embassy .

On a scale of 1-50, how much lexical diversity score do you think this set gets?

41

Set 2:

1.the judges will be here by the next day .

2.i ' il just go to the movies .

3.so , she ' d be happy to be able to communicate with her .

4.we have to go .

5.i ' il give you $ 50 .

6.we got a problem .

7.we ' il be all right .

8.i ' il bet he will .

9.how could he have been involved with the computer ?

10.i ' m not sure .

11.but i was still alive .

12.i ' m not finished .

13.i ' m here to see you .

14.she was at the scene .

15.i ' il take care of it .

On a scale of 1-50, how much lexical diversity score do you think this set gets?

29

Set 3:

1.and they will show up to you , and you will be back in a few minutes .

2.i ' m not sure .

3.the word is kateina , to have seen the kates .

4.we have to go to war .

5.i ' il take it .

6.we ' re in the same area .

7.i ' m gon na have some fun .

8.i ' m sure he ' il have a horse .

9.what kind of files ?

10.i ' m not a bad person .

11.i thank you , mr . bond .

12.i ' m not sure i ' m not gon na do it .

13.i ' m here to see you .

14.they ' re not in charge of this investigation .

15.i ' m going to kill you all .

On a scale of 1-50, how much lexical diversity score do you think this set gets?

32

Figure 3: Interface of Human Evaluation

How reparametrization trick broke differentially-private text representation learning

Ivan Habernal
Trustworthy Human Language Technologies
Department of Computer Science
Technical University of Darmstadt
ivan.habernal@tu-darmstadt.de
www.trusthlt.org

Abstract

As privacy gains traction in the NLP community, researchers have started adopting various approaches to privacy-preserving methods. One of the favorite privacy frameworks, differential privacy (DP), is perhaps the most compelling thanks to its fundamental theoretical guarantees. Despite the apparent simplicity of the general concept of differential privacy, it seems non-trivial to get it right when applying it to NLP. In this short paper, we formally analyze several recent NLP papers proposing text representation learning using DPText (Beigi et al., 2019a,b; Alnasser et al., 2021; Beigi et al., 2021) and reveal their false claims of being differentially private. Furthermore, we also show a simple yet general empirical sanity check to determine whether a given implementation of a DP mechanism almost certainly violates the privacy loss guarantees. Our main goal is to raise awareness and help the community understand potential pitfalls of applying differential privacy to text representation learning.

1 Introduction

Differential privacy (DP), a formal mathematical treatment of privacy protection, is making its way to NLP (Igamberdiev and Habernal, 2021; Senge et al., 2021). Unlike other approaches to protect privacy of individuals' text documents, such as redacting named entities (Lison et al., 2021) or learning text representation with a GAN attacker (Li et al., 2018), DP has the advantage of *quantifying* and *guaranteeing* how much privacy can be lost in the worst case. However, as Habernal (2021) showed, adapting DP mechanisms to NLP properly is a non-trivial task.

Representation learning with protecting privacy in an end-to-end fashion has been recently proposed in DPText (Beigi et al., 2019b,a; Alnasser et al., 2021). DPText consists of an auto-encoder for text representation, a differential-privacy-based noise adder, and private attribute discriminators, among others. The latent text representation is claimed to be differentially private and thus can be shared with data consumers for a given down-stream task. Unlike using a pre-determined privacy budget ε, DPText takes ε as a learnable parameter and utilizes the reparametrization trick (Kingma and Welling, 2014) for random sampling. However, the downstream task results look too good to be true for such low ε values. We thus asked whether DPText is really differentially private.

This paper makes two important contributions to the community. First, we formally analyze the heart of DPText and prove that the employed reparametrization trick based on inverse continuous density function in DPText is wrong and the model violates the DP guarantees. This shows that extreme care should be taken when implementing DP algorithms in end-to-end differentiable deep neural networks. Second, we propose an empirical sanity check which simulates the actual privacy loss on a carefully crafted dataset and a reconstruction attack. This supports our theoretical analysis of non-privacy of DPText and also confirms previous findings of breaking privacy of another system ADePT.[1]

2 Differential privacy primer

Suppose we have a dataset (database) where each element belongs to an individual, for example Alice, Bob, Charlie, up to m. Each person's entry, denoted with a generic variable x, could be an arbitrary object, but for simplicity consider it a real valued vector $x \in \mathbb{R}^k$. An important premise is that this vector contains some sensitive information we aim to protect, for example an income ($x \in \mathbb{R}$), a binary value whether or not the person

[1] ADePT is a text-to-text rewriting system claimed to be differentially private (Krishna et al., 2021) but has been found to be DP-violating (Habernal, 2021).

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 771 - 777
May 22-27, 2022 ©2022 Association for Computational Linguistics

has a certain disease ($x \in \{0.0, 1.0\}$), or a dense representation from SentenceBERT containing the person's latest medical record ($x \in \mathbb{R}^k$). This dataset is held by someone we trust to protect the information, the trusted curator.[2]

This dataset is a set from which we can create 2^m subsets, for instance $X_1 = \{\text{Alice}\}$, $X_2 = \{\text{Alice}, \text{Bob}\}$, etc. All these subsets form a *universe* $\mathcal{X}$, that is $X_1, X_2, \dots \in \mathcal{X}$, and each of them is also called (a bit ambiguously) a dataset.

Definition 2.1. *Any two datasets $X, X' \in \mathcal{X}$ are called* neighboring, *if they differ in one person.*

For example, $X = \{\text{Alice}\}, X' = \{\text{Bob}\}$ or $X = \{\text{Alice}, \text{Bob}\}, X' = \{\text{Bob}\}$ are neighboring, while $X = \{\text{Alice}\}, X' = \{\text{Alice, Bob, Charlie}\}$ are not.

Definition 2.2. Numeric query *is any function f applied to a dataset X and outputting a real-valued vector, formally $f : X \to \mathbb{R}^k$.*

For example, numeric queries might return an average income ($f \to \mathbb{R}$), number of persons in the database ($f \to \mathbb{R}$), or a textual summary of medical records of all persons in the database represented as a dense vector ($f \to \mathbb{R}^k$). The query is simply something we want to learn from the dataset. A query might be also an identity function that just 'copies' the input, e.g., $f(X = \{(1,0)\}) \to (1,0)$ for a real-valued dataset $X = \{(1,0)\}$.

An attacker who knows everything about Bob, Charlie, and others would be able to reveal Alice's private information by querying the dataset and combining it with what they know already. Differentially private algorithm (or mechanism) $\mathcal{M}(X; f)$ thus randomly modifies the query output in order to minimize and quantify such attacks. Smith and Ullman (2021) formulate the principle of differential privacy as follows: *"No matter what they know ahead of time, an attacker seeing the output of a differentially private algorithm would draw (almost) the same conclusions about Alice whether or not her data were used."*

Let a DP-mechanism $\mathcal{M}(X; f)$ have an arbitrary range $\mathcal{R}$ (a generalization of our case of numeric queries, for which we would have $\mathcal{R} = \mathbb{R}^k$). Differential privacy is then defined as

$$\frac{\Pr(X|\mathcal{M}(X;f) = z)}{\Pr(X'|\mathcal{M}(X;f) = z)} \leq \exp(\varepsilon) \cdot \frac{\Pr(X)}{\Pr(X')} \quad (1)$$

for all neighboring datasets X, X' and all $z \in \mathcal{R}$, where $\Pr(X)$ and $\Pr(X')$ is our prior knowledge of X and X'. In words, our posterior knowledge of X or X' after observing z can only grow by factor $\exp(\varepsilon)$ (Mironov, 2017), where ε is a *privacy budget* (Dwork and Roth, 2013).[3]

3 Analysis of DPText

In the heart of the model, DPText relies on the standard Laplace mechanism which takes a real-valued vector and perturbs each element by a random draw from the Laplace distribution.

Formally, let $\mathbf{z}$ be a real-valued d-dimensional vector. Then the Laplace mechanism outputs a vector $\tilde{\mathbf{z}}$ such that for each index $i = 1, \dots, d$

$$\tilde{z}_i = z_i + s_i \quad (2)$$

where each s_i is drawn independently from a Laplace distribution with zero mean and scale b that is proportional to the ℓ_1 sensitivity Δ and the privacy budget ε, namely

$$s_i \sim \text{Lap}\left(\mu = 0; b = \frac{\Delta}{\varepsilon}\right) \quad (3)$$

The Laplace mechanism satisfies differential privacy (Dwork and Roth, 2013).

3.1 Reparametrization trick and inverse CDF sampling

DPText employs the variational autoencoder architecture in order to directly optimize the amount of noise added in the latent layer parametrized by ε. In other words, the scale of the Laplace distribution becomes a trainable parameter of the network. As directly sampling from a distribution is known to be problematic for end-to-end differentiable deep networks, DPText borrows the reparametrization trick from Kingma and Welling (2014).

In a nutshell, the reparametrization trick decouples drawing a random sample from a desired distribution (such as Exponential, Laplace, or Gaussian) into two steps: First draw a value from another distribution (such as Uniform), and then transform it using a particular function, mainly the inverse continuous density function (CDF).

As a matter of fact, sampling using the inverse CDF is a well-known and widely used

[2] This is *centralized DP*, as opposed to *local-DP* where no such trusted curator exists.

[3] In this paper, we will use the basic form of DP, that is $(\varepsilon, 0)$-DP. There are various other (typically more 'relaxed') variants of DP, such (ε, δ)-DP, but they are not relevant to the current paper as DPText also claims $(\varepsilon, 0)$-DP.

method (Devroye, 1986; Ross, 2012) and forms the backbone of probability distribution generators in many popular frameworks.

3.2 Inverse CDF of Laplace distribution

The inverse cumulative distribution function of Laplace distribution $\text{Lap}(\mu; b)$ is:

$$F^{-1}(u) = \mu - b \, \text{sgn}(u - 0.5) \, \ln(1 - 2|u - 0.5|) \tag{4}$$

where $u \sim \text{Uni}(0, 1)$ is drawn from a standard uniform distribution (Sugiyama, 2016, p. 210), (Nahmias and Olsen, 2015, p. 303). An equivalent expression without the sgn and absolute functions is derived, e.g., by Li et al. (2019, p. 166) as

$$F^{-1}(u) = \begin{cases} b \ln(2u) + \mu & \text{if } u < 0.5 \\ \mu - b \ln(2(1 - u)) & \text{if } u \geq 0.5 \end{cases} \tag{5}$$

where again $u \sim \text{Uni}(0, 1)$.[4]

An alternative sampling strategy, as shown, e.g., by Al-Shuhail and Al-Dossary (2020, p. 62), assumes that the random variable is drawn from a shifted, zero-centered uniform distribution

$$v \sim \text{Uni}\,(-0.5, +0.5) \tag{6}$$

and transformed through the following function

$$F^{-1}(v) = \mu - b \, \text{sgn}(v) \ln(1 - 2|v|) \tag{7}$$

While both (4) and (7) generate samples from $\text{Lap}(\mu; b)$, note the substantial difference between u and v, since each is drawn from a different uniform distribution (see visualizations in Fig. 1).

3.3 Proofs of DPText violating DP

According to Eq. 3 in (Alnasser et al., 2021), Eq. 9 in (Beigi et al., 2019a) which is an extended version of (Beigi et al., 2019b), in Eq. 14 in (Beigi et al., 2021), and personal communication to confirm the formulas, the main claim of DPText is as follows (rephrased):

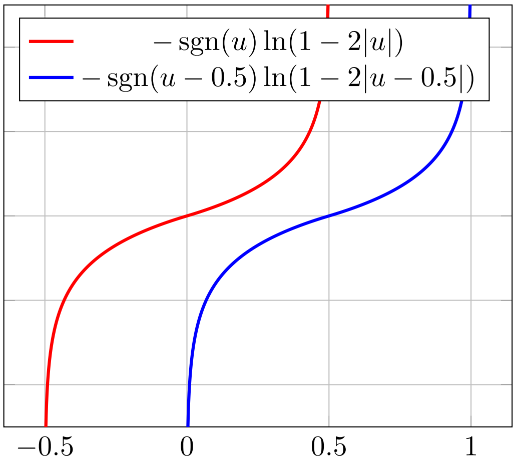


Figure 1: Inverse CDFs for Laplace sampling.

> DPText utilizes the Laplace mechanism, which is DP (Dwork and Roth, 2013). It implements the mechanism as follows: Sampling a value from standard uniform
>
> $$v \sim \text{Uni}(0, 1) \tag{8}$$
>
> and transforming using
>
> $$F^{-1}(v) = \mu - b \, \text{sgn}(v) \ln(1 - 2|v|) \tag{9}$$
>
> is equivalent to sampling noise from $\text{Lap}(b)$.

This claim is unfortunately false, as it mixes up both approaches introduced in Sec. 3.2. As a consequence, the Laplace mechanism using such sampling is not DP, which we will first prove formally.

Theorem 3.1. *Sampling using inverse CDF as in DPText using (8) and (9) does not produce Laplace distribution.*

Proof. We will rely on the standard proof of sampling from inverse CDF (see Appendix A). The essential step of that proof is that the CDF is increasing on the support of the uniform distribution, that is on $[0, 1]$. However, F^{-1} as used in (9) is increasing only on interval $[0, 0.5]$ (Fig. 1). For $v \geq 0.5$, we get negative argument to ln which yields a complex function, whose real part is even decreasing. Therefore (9) is not CDF of any probability distribution, if used with $\text{Uni}(0, 1)$. □

As a consequence, the output $\ln(v \leq 0)$ arbitrarily depends on the particular implementation. In `numpy`, it is `NaN` with a warning only. Therefore this function samples only positive or `NaN` numbers. Since DPText sources are not publicly available, we can only assume that `NaN` numbers

[4] This implementation is used in `numpy`, see https://github.com/numpy/numpy/blob/maintenance/1.21.x/numpy/random/src/distributions/distributions.c#L469

are either replaced by zero, or the sampling proceeds as long as the desired number of samples is reached (discarding `NaN`s). In either case, no negative values can be obtained. See Fig. 3 in the Appendix for various Laplace-based distributions sampled with different techniques including possible distributions sampled in DPText.

Theorem 3.2. *DPText with private mechanism based on (8) and (9) fails to guarantee differential privacy.*

Proof. We rely on the standard proof of the Laplace mechanism as shown, e.g, by Habernal (2021). Let $X = 0$ and $X' = 1$ be two neighboring datasets, and the query f being the identity query, such that it outputs simply the value of X. Let the DPText mechanism $\mathcal{M}(X; f)$ outputs a particular value z.

In order to being differentially private, mechanism $\mathcal{M}(X; f)$ has to fulfill the following bound of the privacy loss:

$$\left| \frac{\Pr(\mathcal{M}(X) = z)}{\Pr(\mathcal{M}(X') = z)} \right| \leq \exp(\varepsilon) \qquad (10)$$

for all neighboring datasets $X, X' \in \mathcal{X}$ and all outputs $z \in \mathcal{R}$ from the range of $\mathcal{M}$, provided that our priors over X and X' are uniform (cf. Eq. 1).

Fix $z = 0.1$. Then $\Pr(\mathcal{M}(X) = 0.1)$ will have a positive probability (recall it takes the query output $f(X = 0) = 0$ and adds a random number drawn from the probability distribution, which is always positive as shown in Theorem 3.1.) However $\Pr(\mathcal{M}(X') = 0.1)$ will be zero, as the query output $f(X' = 1) = 1$ will be added again only a positive random number and thus never be less than 1. By plugging this into (10), we obtain

$$\left| \frac{\Pr(\mathcal{M}(X) = 0.1)}{\Pr(\mathcal{M}(X') = 0.1)} \right| = \frac{\Pr > 0}{\Pr = 0} \nleq \exp(\varepsilon) \quad (11)$$

which results in an infinity privacy loss and violates differential privacy. □

4 Empirical sanity check algorithm

It is impossible to empirically verify that a given DP-mechanism implementation is actually DP (Ding et al., 2018). However, it is possible to detect a DP-violating mechanism with a fair degree of certainty. We propose a general sanity check applicable to any real-valued DP mechanism, such as the Laplace mechanism, DPText, or any other.[5]

We start by constructing two neighboring datasets X (Alice) and X' (Bob) such that $X = (0, \dots, 0_n)$ consists of n zeros and $X' = (1, \dots, 1_n)$ consists of n ones. The dimensionality $n \in \{1, 2, \dots\}$ is a hyperparameter of the experiment. We employ a synthetic data release mechanism (also called local DP). The mechanism takes X or X' and outputs its privatized version of the same dimensionality n, so that the zeros or ones are 'noisified' real numbers. The query sensitivity Δ is n.[6]

Thanks to the post-processing lemma, any post-processing of DP output remains DP. We can thus turn the output real vector back to all zeros or all ones, simply by rounding to closest 0 or 1 and applying majority voting. This process is in fact our reconstruction attack: given a privatized vector, we try to guess what the original values were, either all zeros or all ones.

What our attacker is doing, and what DP protects, is that if Alice gives us her privatized data, we cannot tell whether her private values were all zeros or all ones (up to a given factor); the same for Bob.

By definition (1) and having no prior knowledge over X and X' apart from the fact that the values are correlated, our attacker cannot exceed the guaranteed privacy loss $\exp(\varepsilon)$:

$$\frac{\Pr(X | \mathcal{M}(X; f) = z)}{\Pr(X' | \mathcal{M}(X; f) = z)} \leq \exp(\varepsilon) \qquad (12)$$

We can estimate the conditional probability $\Pr(X | \mathcal{M}(X; f) = z)$ using maximum likelihood estimation (MLE) simply as our attacker's precision: How many times the attacker reconstructed true X values given the observed privatized vector. We can do the same for estimating the conditional probability of X'. In particular, we repeatedly run each DP mechanism over X and X' 10 million times each, which gives very precise MLE estimates even for small ε.[7]

[5] Some related works along these lines also utilize statistical analysis of the source code written in a C-like language (Wang et al., 2020).

[6] See (Dwork and Roth, 2013) for ℓ_1-sensitivity definition.

[7] For example, we repeated the full experiment on ADePT ($n = 2$, $\varepsilon = 0.1$) 100 times which results in standard deviation 0.0008 from the mean value 0.195. Better MLE precision can be simply obtained by increasing the 10 million repeats per experiment. Source codes available at `https://github.com/trusthlt/`

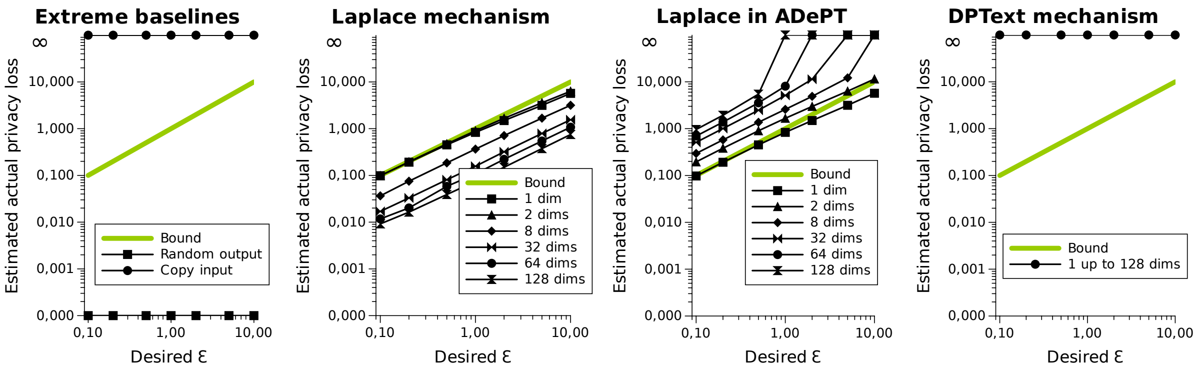


Figure 2: Area under the green line: Our attack does not reveal more than allowed by the desired privacy budget. Note that it does not guarantee DP, the reconstruction attack might be just weak. Area above the green line: The algorithm almost certainly violates DP as our attack caused bigger privacy loss than allowed by ε. *Extreme baselines* show two extreme scenarios, as *random output* is absolutely private (but provides zero utility) and *copy input* provides maximal utility but no privacy by revealing the data in full.

5 Results and discussion

For the sake of completeness, we implemented two extreme baselines: One that simply copies input (no privacy) and other one completely random regardless of the input (maximum privacy); these are shown in Figure 2 left. The vanilla Laplace mechanism behaves as expected; all empirical losses for all dimensions (1 up to 128) are bounded by ε. We re-implemented the Laplace mechanism from ADePT (Krishna et al., 2021) that, due to wrong sensitivity, has been shown theoretically as DP-violating (Habernal, 2021). We empirically confirm that ADePT suffered from the curse of dimensionality as the privacy loss explodes for larger dimensions. The last panel confirms our previous theoretical DPText results, which (regardless of dimensionality) has infinite privacy loss.

Note that we constructed the dataset carefully as two neighboring multidimensional correlated data that are as distant from each other as possible in the $(0, 1)^n$ space. However, DP must guarantee privacy for any datapoints, even the worst case scenario, as shown by the correct Laplace mechanism.

6 Conclusion

We formally proved that DPText (Beigi et al., 2019b,a; Alnasser et al., 2021; Beigi et al., 2021) is not differentially private due to wrong sampling in its reparametrization trick. We also proposed an empirical sanity check that confirmed our findings and can help to reveal potential errors in DP mechanism implementations for NLP.

7 Ethics Statement

We declare no conflict of interests with the authors of DPText, we do not even know them personally. The purpose of this paper is strictly scientific.

Acknowledgements

The independent research group TrustHLT is supported by the Hessian Ministry of Higher Education, Research, Science and the Arts. Thanks to Cecilia Liu, Haau-Sing Li, and the anonymous reviewers for their helpful feedback. A special thanks to Condor airlines, whose greed to make passengers pay for everything resulted in the most productive transatlantic flights I've ever had.

References

Abdullatif Al-Shuhail and Saleh Al-Dossary. 2020. *Robust Filter—Dealing with Impulse Noise*, pages 61–80. Springer International Publishing.

Walaa Alnasser, Ghazaleh Beigi, and Huan Liu. 2021. Privacy Preserving Text Representation Learning Using BERT. In *Proceedings of the 14th International Conference on Social, Cultural, and Behavioral Modeling (SBP-BRiMS)*, pages 91–100, Virtual event. Springer International Publishing.

John E. Angus. 1994. The Probability Integral Transform and Related Results. *SIAM Review*, 36(4):652–654.

acl2022-reparametrization-trick-broke-differential-privacy

Ghazaleh Beigi, Kai Shu, Ruocheng Guo, Suhang Wang, and Huan Liu. 2019a. I Am Not What I Write: Privacy Preserving Text Representation Learning. *arXiv preprint*.

Ghazaleh Beigi, Kai Shu, Ruocheng Guo, Suhang Wang, and Huan Liu. 2019b. Privacy Preserving Text Representation Learning. In *Proceedings of the 30th ACM Conference on Hypertext and Social Media*, pages 275–276, Hof, Germany. ACM.

Ghazaleh Beigi, Kai Shu, Ruocheng Guo, Suhang Wang, and Huan Liu. 2021. Systems and methods for a privacy preserving text representation learning framework. U.S. Patent, Pending Application US20210342546A1, Application filed by Arizona Board of Regents of ASU.

Luc Devroye. 1986. *Non-uniform random variate generation*. Springer-Verlag New York Inc.

Zeyu Ding, Yuxin Wang, Guanhong Wang, Danfeng Zhang, and Daniel Kifer. 2018. Detecting Violations of Differential Privacy. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 475–489, Toronto, Canada. ACM.

Cynthia Dwork and Aaron Roth. 2013. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3-4):211–407.

Ivan Habernal. 2021. When differential privacy meets NLP: The devil is in the detail. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1522–1528, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Timour Igamberdiev and Ivan Habernal. 2021. Privacy-Preserving Graph Convolutional Networks for Text Classification. *arXiv preprint*.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, pages 1–14, Banff, Canada.

Satyapriya Krishna, Rahul Gupta, and Christophe Dupuy. 2021. ADePT: Auto-encoder based Differentially Private Text Transformation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2435–2439, Online. Association for Computational Linguistics.

Xianxian Li, Huaxing Zhao, Dongran Yu, Li-e Wang, and Peng Liu. 2019. Multidimensional Correlation Hierarchical Differential Privacy for Medical Data with Multiple Privacy Requirements. In *Proceedings of the 2nd International Conference on Healthcare Science and Engineering*, pages 153–173, Guilin, China. Springer Singapore.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards Robust and Privacy-preserving Text Representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30, Melbourne, Australia. Association for Computational Linguistics.

Pierre Lison, Ildikó Pilán, David Sanchez, Montserrat Batet, and Lilja Øvrelid. 2021. Anonymisation Models for Text Data: State of the art, Challenges and Future Directions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4188–4203, Online. Association for Computational Linguistics.

Ilya Mironov. 2017. Rényi Differential Privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275, Santa Barbara, piiCA, USA. IEEE.

Steven Nahmias and Tava Lennon Olsen. 2015. *Production and Operations Analysis*, 7th edition. Waveland Press, Inc.

Sheldon Ross. 2012. *Simulation*, 5th edition. Academic Press.

Manuel Senge, Timour Igamberdiev, and Ivan Habernal. 2021. One size does not fit all: Investigating strategies for differentially-private learning across NLP tasks. *arXiv preprint*.

Adam Smith and Jonathan Ullman. 2021. Privacy in Statistics and Machine Learning. Lecture 5: Differential Privacy II.

Masashi Sugiyama. 2016. *Introduction to Statistical Machine Learning*. Morgan Kaufmann.

Yuxin Wang, Zeyu Ding, Daniel Kifer, and Danfeng Zhang. 2020. CheckDP: An Automated and Integrated Approach for Proving Differential Privacy or Finding Precise Counterexamples. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 919–938, Online. ACM.

A Proof of sampling from inverse CDF

Important fact 1: A random variable U is uniformly distributed on $[0, 1]$ if the following holds

$$U \sim \text{Uni}(0, 1) \iff \Pr(U \leq u) = u. \tag{13}$$

Important fact 2: For any function $g(\cdot)$ with an inverse function $g^{-1}(\cdot)$, the following holds

$$g(g^{-1}(x)) = x; \quad g^{-1}(g(x)) = x. \tag{14}$$

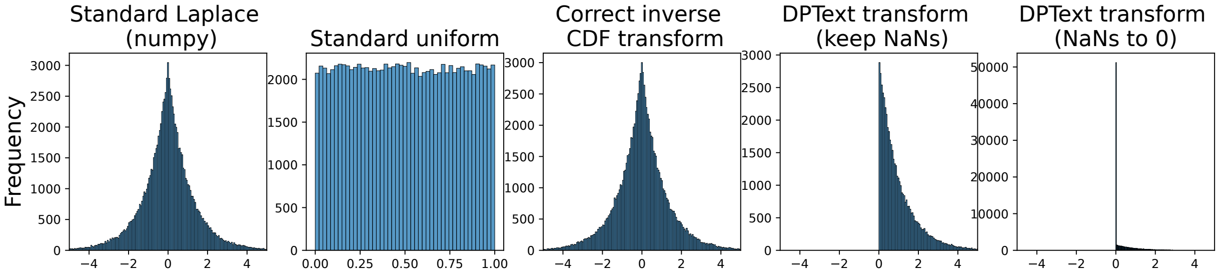


Figure 3: Comparing sampling strategies. Left: Sampling using vanilla `numpy` implementation. Second from the left: Uniform sample as basis for the following three inverse CDF transformations. Generated with 100k samples.

Important fact 3: For any increasing function $g(\cdot)$, we have by definition

$$x \leq y \implies g(x) \leq g(y). \tag{15}$$

We know that $\Pr(X \leq a)$ is a shortcut for probability of event E_1 defined using the set-builder notation as $E_1 = \{s \in \Omega : X(s) \leq a\}$. Then by plugging (15) into the predicate of E_1, we obtain an equal set, namely event $E_2 = \{s \in \Omega : g(X(s)) \leq g(a)\}$, for which the probability must be the same. Therefore for any random variable X and increasing function $g(\cdot)$ we have

$$\Pr(X \leq a) = \Pr(g(X) \leq g(a)). \tag{16}$$

Theorem A.1. *Let U be a uniform random variable on $[0, 1]$. Let X be a continuous random variable with CDF (cumulative distribution function) $F(\cdot)$. Let Y be defined such that $Y = F^{-1}(U)$. Then Y has CDF $F(\cdot)$.*

Proof. Function $F(\cdot)$ is the CDF of a continuous random variable X, and as a CDF its range is $[0, 1]$. Also, if $F(\cdot)$ is strictly increasing, it has a unique inverse function $F^{-1}(\cdot)$ defined on $[0, 1]$.

We defined $Y = F^{-1}(U)$, so consider

$$\Pr(Y \leq y) = \Pr\big(F^{-1}(U) \leq y\big). \tag{17}$$

Since $F(\cdot)$ is increasing, using (16) we get

$$\Pr(Y \leq y) = \Pr\big(F(F^{-1}(U)) \leq F(y)\big). \tag{18}$$

Now plugging (14) we obtain

$$\Pr(Y \leq y) = \Pr(U \leq F(y)), \tag{19}$$

and finally by (13)

$$\Pr(Y \leq y) = F(y). \tag{20}$$

□

For an overview of proofs of Theorem A.1 see (Angus, 1994).

Towards Consistent Document-level Entity Linking: Joint Models for Entity Linking and Coreference Resolution

Klim Zaporojets, Johannes Deleu, Yiwei Jiang, Thomas Demeester, Chris Develder
Ghent University – imec, IDLab
Ghent, Belgium
{first_name.last_name}@ugent.be

Abstract

We consider the task of document-level entity linking (EL), where it is important to make consistent decisions for entity mentions over the full document jointly. We aim to leverage explicit "connections" among mentions within the document itself: we propose to join EL and coreference resolution (coref) in a *single* structured prediction task over directed trees and use a globally normalized model to solve it. This contrasts with related works where two separate models are trained for each of the tasks and additional logic is required to merge the outputs. Experimental results on two datasets show a boost of up to +5% F1-score on both coref and EL tasks, compared to their standalone counterparts. For a subset of hard cases, with individual mentions lacking the correct EL in their candidate entity list, we obtain a +50% increase in accuracy.[1]

1 Introduction

In this paper we explore a principled approach to solve entity linking (EL) jointly with coreference resolution (coref). Concretely, we formulate coref+EL as a *single* structured task over directed trees that conceives EL and coref as two complementary components: a coreferenced cluster can only be linked to a single entity or NIL (i.e., a non-linkable entity), and all mentions linking to the same entity are coreferent. This contrasts with previous attempts to join coref+EL (Hajishirzi et al., 2013; Dutta and Weikum, 2015; Angell et al., 2021) where coref and EL models are trained separately and additional logic is required to merge the predictions of both tasks.

Our first approach (Local in Fig. 1(a)) is motivated by current state-of-the-art coreference resolution models (Joshi et al., 2019; Wu et al., 2020) that predict a single antecedent for each span to resolve.

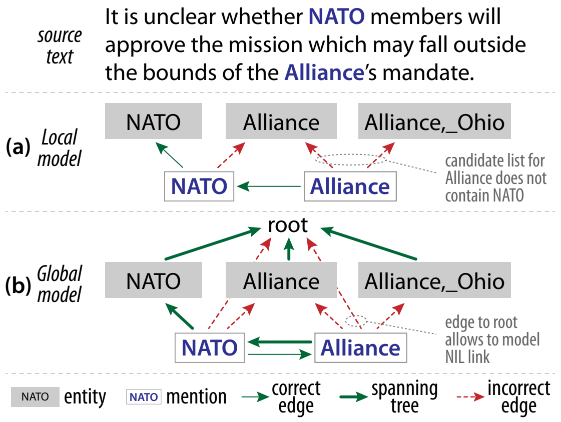

source text
It is unclear whether NATO members will approve the mission which may fall outside the bounds of the Alliance's mandate.
(a) Local model
NATO
Alliance
Alliance,_Ohio
NATO
Alliance
candidate list for Alliance does not contain NATO
root
(b) Global model
NATO
Alliance
Alliance,_Ohio
NATO
Alliance
edge to root allows to model NIL link
NATO entity
NATO mention
correct edge
spanning tree
incorrect edge

Figure 1: Illustration of our 2 explored graph models: (a) Local where edges are only allowed from spans to antecedents or candidate entities, and (b) Global where the prediction involves a spanning tree over all nodes.

We extend this architecture by also considering entity links as potential antecendents: in the example of Fig. 1, the mention "Alliance" can be either connected to its antecedent mention "NATO" or to any of its candidate links (*Alliance* or *Alliance,_Ohio*). While straightforward, this approach cannot solve cases where the first coreferenced mention does not include the correct entity in its candidate list (e.g., if the order of "NATO" and "Alliance" mentions in Fig. 1 would be reversed). We therefor propose a second approach, Global, which by construction overcomes this inherent limitation by using bidirectional connections between mentions. Because that implies cycles could be formed, we resort to solving a maximum spanning tree problem. Mentions that refer to the same entity form a cluster, represented as a subtree rooted by the single entity they link to. To encode the overall document's clusters in a single spanning tree, we introduce a virtual *root* node (see Fig. 1(b)).[2]

This paper contributes: (i) 2 architectures (Local and Global) for joint entity linking (EL) and

[1] Our code, models and AIDA$^+$ dataset will be released on https://github.com/klimzaporojets/consistent-EL

[2] Coreference clusters without a linked entity, i.e., a NIL cluster, have a link of a mention directly to the root.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 778 - 784
May 22-27, 2022 ©2022 Association for Computational Linguistics

corefence resolution, (ii) an extended AIDA dataset (Hoffart et al., 2011), adding new annotations of linked and NIL coreference clusters, (iii) experimental analysis on 2 datasets where our joint coref+EL models achieve up to +5% F1-score on both tasks compared to standalone models. We also show up to +50% in accuracy for hard cases of EL where entity mentions lack the correct entity in their candidate list.

2 Architecture

Our model takes as input (i) the full document text, and (ii) an *alias table* with entity candidates for each of the possible spans. Our end-to-end approach allows to jointly predict the mentions, entity links and coreference relations between them.

2.1 Span and Entity Representations

We use SpanBERT (base) from Joshi et al. (2020) to obtain *span* representations $\mathbf{g}_i$ for a particular span s_i. Similarly to Luan et al. (2019); Xu and Choi (2020), we apply an additional pruning step to keep only the top-N spans based on the pruning score Φ_p from a feed-forward neural net (FFNN):

$$\Phi_\mathrm{p}(s_i) = \mathrm{FFNN}_P(\mathbf{g}_i). \tag{1}$$

For a candidate entity e_j of span s_i we will obtain representation as $\mathbf{e}_j$ (which is further detailed in §3).

2.2 Joint Approaches

We propose two methods for joint coreference and EL. The first, Local, is motivated by end-to-end span-based coreference resolution models (Lee et al., 2017, 2018) that optimize the marginalized probability of the correct antecedents for each given span. We extend this local marginalization to include the span's candidate entity links. Formally, the modeled probability of y (text span or candidate entity) being the antecedent of span s_i is:

$$P_\mathrm{cl}(y|s_i) = \frac{\exp\left(\Phi_\mathrm{cl}(s_i, y)\right)}{\sum_{y' \in \mathcal{Y}(s_i)} \exp\left(\Phi_\mathrm{cl}(s_i, y')\right)}, \tag{2}$$

where $\mathcal{Y}(s_i)$ is the set of antecedent spans unified with the candidate entities for s_i. For antecedent *spans* $\{s_j : j < i\}$ the score Φ_cl is defined as:

$$\Phi_\mathrm{cl}(s_i, s_j) = \Phi_\mathrm{p}(s_i) + \Phi_\mathrm{p}(s_j) + \Phi_\mathrm{c}(s_i, s_j), \tag{3}$$

$$\Phi_\mathrm{c}(s_i, s_j) = \mathrm{FFNN}_C([\mathbf{g}_i; \mathbf{g}_j; \mathbf{g}_i \odot \mathbf{g}_j; \boldsymbol{\varphi}_{i,j}]), \tag{4}$$

where $\boldsymbol{\varphi}_{i,j}$ is an embedding encoding the distance[3] between spans s_i and s_j. Similarly, for a particular candidate *entity* e_j, the score Φ_cl is:

$$\Phi_\mathrm{cl}(s_i, e_j) = \Phi_\mathrm{p}(s_i) + \Phi_\ell(s_i, e_j), \tag{5}$$

$$\Phi_\ell(s_i, e_j) = \mathrm{FFNN}_L([\mathbf{g}_i; \mathbf{e}_j]). \tag{6}$$

An example graph of mentions and entities with edges for which aforementioned scores Φ_cl would be calculated is sketched in Fig. 1(a). While simple, this approach fails to correctly solve EL when the correct entity is only present in the candidate lists of mention spans occurring later in the text (since earlier mentions have no access to it).

To solve EL in the general case, even when the first mention does not have the correct entity, we propose bidirectional connections between mentions, thus leading to a maximum spanning tree problem in our Global approach. Here we define a score for a (sub)tree t, noted as $\Phi_\mathrm{tr}(t)$:

$$\Phi_\mathrm{tr}(t) = \sum_{(i,j) \in t} \Phi_\mathrm{cl}(u_i, u_j), \tag{7}$$

where u_i and u_j are two connected nodes (i.e., *root*, candidate entities or spans) in t. For a ground truth cluster $c \in C$ (with C being the set of all such clusters), with its set[4] of correct subtree representations $\mathcal{T}_c$, we model the cluster's likelihood with its subtree scores. We minimize the negative log-likelihood $\mathcal{L}$ of all clusters:

$$\mathcal{L} = -\log \frac{\prod_{c \in C} \sum_{t \in \mathcal{T}_c} \exp\left(\Phi_\mathrm{tr}(t)\right)}{\sum_{t \in \mathcal{T}_{all}} \exp\left(\Phi_\mathrm{tr}(t)\right)}. \tag{8}$$

Naively enumerating all possible spanning trees ($\mathcal{T}_{all}$ or $\mathcal{T}_c$) implied by this equation is infeasible, since their number is exponentially large. We use the adapted Kirchhoff's Matrix Tree Theorem (MTT; Koo et al. (2007); Tutte (1984)) to solve this: the sum of the weights of the spanning trees in a directed graph rooted in *r* is equal to the determinant of the Laplacian matrix of the graph with the row and column corresponding to *r* removed (i.e., the *minor* of the Laplacian with respect to *r*). This way, eq. (8) can be rewritten as

$$\mathcal{L} = -\log \frac{\prod_{c \in C} \det\left(\hat{\mathbf{L}}_c(\boldsymbol{\Phi}_\mathrm{cl})\right)}{\det\left(\mathbf{L}_r(\boldsymbol{\Phi}_\mathrm{cl})\right)}, \tag{9}$$

[3] Measured in number of spans, after pruning.

[4] For a single cluster annotation, indeed it is possible that multiple correct trees can be drawn.

where $\mathbf{\Phi}_{\text{cl}}$ is the weighted adjacency matrix of the graph, and $\mathbf{L}_r$ is the minor of the Laplacian with respect to the root node r. An entry in the Laplacian matrix is calculated as

$$L_{i,j} = \begin{cases} \sum\limits_{k} \exp(\Phi_{\text{cl}}(u_k, u_j)) & \text{if } i = j \\ -\exp(\Phi_{\text{cl}}(u_i, u_j)) & \text{otherwise} \end{cases}, \tag{10}$$

Similarly, $\hat{\mathbf{L}}_c$ is a *modified Laplacian* matrix where the first row is replaced with the root r selection scores $\Phi_{\text{cl}}(r, u_j)$. For clarity, Appendix A presents a toy example with detailed steps to calculate the loss in eq. (9).

To calculate the scores of each of the entries $\Phi_{\text{cl}}(u_i, u_j)$ to $\mathbf{\Phi}_{\text{cl}}$ matrix in eqs. (7) and (9) for Global, we use the same approach as in Local for edges between two mention spans, or between a mention and entity. For the directed edges between the root r and a candidate entity e_j we choose $\Phi_{\text{cl}}(r, e_j) = 0$. Since we represent NIL clusters by edges from the mention spans directly to the root, we also need scores for them: we use eq. (3) with $\Phi_{\text{p}}(r) = 0$. We use Edmonds' algorithm (Edmonds, 1967) for decoding the maximum spanning tree.

3 Experimental Setup

We considered two datasets to evaluate our proposed models: DWIE (Zaporojets et al., 2021) and AIDA (Hoffart et al., 2011). Since AIDA essentially does not contain coreference information, we had to extend it by (i) adding missing mention links in order to make annotations consistent on the coreference cluster level, and (ii) annotating NIL coreference clusters. We note this extended dataset as AIDA$^+$. See Table 1 for the details.

As input to our models, for DWIE we generate spans of up to 5 tokens. For each mention span s_i, we find candidates from a dictionary of entity surface forms used for hyperlinks in Wikipedia. We then keep the top-16 candidates based on the prior for that surface form, as per Yamada et al. (2016, §3). Each of those candidates e_j is represented using a Wikipedia2Vec embedding $\mathbf{e}_j$ (Yamada et al., 2016).[5] For AIDA$^+$, we use the spans, entity candidates, and entity representations from Kolitsas et al. (2018).[6]

To assess the performance of our joint coref+EL models Local and Global, we also provide Standalone implementations for coref and EL tasks. The Standalone coref model is trained using only the coreference component of our joint architecture (eq. (2)–(4)), while the EL model is based only on the linking component (eq. (6)).

Dataset	# Linked clusters	# NIL clusters	Linked mentions	# NIL mentions
DWIE	11,967	9,935	28,482	14,891
AIDA	16,673	-	27,817	7,112
AIDA$^+$	16,775	4,284	28,813	6,116

Table 1: Datasets statistics.

As performance metrics, for coreference resolution we calculate the average-F1 score of commonly used MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998) and CEAF$_\text{e}$ (Luo, 2005) metrics as implemented by Pradhan et al. (2014). For EL, we use (i) *mention*-level F1 score (EL$_\text{m}$), and (ii) *cluster*-level *hard* F1 score (EL$_\text{h}$) that counts a true positive only if both the coreference cluster (in terms of all its mention spans) and the entity link are correctly predicted. These EL metrics are executed in a *strong matching* setting that requires predicted spans to exactly match the boundaries of gold mentions. Furthermore, for EL we only report the performance on non-NIL mentions, leaving the study of NIL links for future work.

Our experiments will answer the following research questions: **(Q1)** How does performance of our joint coref+EL models compare to Standalone models? **(Q2)** Does jointly solving coreference resolution and EL enable more coherent EL predictions? **(Q3)** How do our joint models perform on hard cases where some individual entity mentions do not have the correct candidate?

4 Results

Table 2 shows the results of our compared models for EL and coreference resolution tasks. Answering **(Q1)**, we observe a general improvement in performance of our coref+EL joint models (Local and Global) compared to Standalone on the EL task. Furthermore, this difference is bigger when using our cluster-level *hard* metrics. This also answers **(Q2)** by indicating that the joint models tend to produce more coherent cluster-based predictions. To make this more explicit, Table 3 compares the accuracy for singleton clusters (i.e., clusters composed by a single entity mention), denoted as S, to that of clusters composed by multiple mentions, denoted

[5] We use Wikipedia version 20200701.

[6] https://github.com/dalab/end2end_neural_el

	DWIE			AIDA^+_a			AIDA^+_b		
Setup	EL$_m$	EL$_h$	Coref	EL$_m$	EL$_h$	Coref	EL$_m$	EL$_h$	Coref
Standalone	88.7±0.1	78.4±0.2	94.5±0.1	86.2±0.4	80.7±0.5	93.8±0.1	79.1±0.3	74.0±0.3	91.5±0.3
Local	90.5±0.4	83.4±0.4	94.4±0.2	87.5±0.2	83.1±0.2	94.7±0.1	**79.9±0.4**	75.8±0.3	**92.3±0.1**
Global	**90.7±0.3**	**83.9**±0.5	**94.7**±0.2	**87.6±0.2**	**83.7**±0.3	**95.1**±0.1	79.6±0.4	**76.0**±0.4	92.2±0.2

Table 2: Experimental results (F1 scores defined in §3) using the Standalone coreference and EL models compared to our joint architectures (Local and Global), on DWIE and AIDA$^+$ datasets.

	DWIE		AIDA^+_a		AIDA^+_b	
Setup	S	M	S	M	S	M
Standalone	80.4	69.5	82.9	70.7	77.0	57.0
Local	**82.6**	78.6	84.9	74.8	**79.8**	61.4
Global	**82.6**	**80.0**	**85.1**	**76.8**	79.3	**63.0**

Table 3: Cluster-based accuracy of link prediction on singletons (S) and clusters of multiple mentions (M).

Setup	DWIE	AIDA^+_a	AIDA^+_b
Standalone	0.0	0.0	0.0
Local	41.7	27.4	26.9
Global	**57.6**	**50.2**	**29.7**

Table 4: EL accuracy for corner case mentions where the correct entity is not in the mention's candidate list.

as M. We observe that the difference in performance between our joint models and Standalone is bigger on M clusters (with a consistent superiority of Global), indicating that our approach indeed produces more coherent predictions for mentions that refer to the same concept. Further analysis reveals that this difference in performance is even higher for a more complex scenario where the clusters contain mentions with different surface forms (not shown in the table).

In order to tackle research question **(Q3)**, we study the accuracy of our models on the important corner case that involves mentions without correct entity in their candidate lists. This is illustrated in Table 4, which focuses on such mentions in clusters where at least one mention contains the correct entity in its candidate list. As expected, the Standalone model cannot link such mentions, as it is limited to the local candidate list. In contrast, both our joint approaches can solve some of these cases by using the correct candidates from other mentions in the cluster, with a superior performance of our Global model compared to the Local one.

5 Related Work

Entity Linking: Related work in entity linking (EL) tackles the document-level linking coherence by exploring relations between entities (Kolitsas et al., 2018; Yang et al., 2019; Le and Titov, 2019), or entities and mentions (Le and Titov, 2018). More recently, contextual BERT-driven (Devlin et al., 2019) language models have been used for the EL task (Broscheit, 2019; De Cao et al., 2020, 2021; Yamada et al., 2020) by jointly embedding mentions and entities. In contrast, we explore a cluster-based EL approach where the coherence is achieved on *coreferent* entity mentions level.

Coreference Resolution: Span-based antecedent-ranking coreference resolution (Lee et al., 2017, 2018) has seen a recent boost by using SpanBERT representations (Xu and Choi, 2020; Joshi et al., 2020; Wu et al., 2020). We extend this approach in our Local joint coref+EL architecture. Furthermore, we rely on Kirchhoff's Matrix Tree Theorem (Koo et al., 2007; Tutte, 1984) to efficiently train a more expressive spanning tree-based Global method.

Joint EL+Coref: Fahrni and Strube (2012) introduce a more expensive rule-based Integer Linear Programming component to jointly predict coref and EL. Durrett and Klein (2014) jointly train coreference and entity linking without enforcing single-entity per cluster consistency. More recently, Angell et al. (2021); Agarwal et al. (2021) use additional logic to achieve consistent cluster-level entity linking. In contrast, our proposed approach constrains the space of the predicted spanning trees on a structural level (see Fig. 1).

6 Conclusion

We propose two end-to-end models to solve entity linking and coreference resolution tasks in a joint setting. Our joint architectures achieve superior performance compared to the standalone counterparts. Further analysis reveals that this boost in performance is driven by more coherent predictions on the level of mention clusters (linking to the same entity) and extended candidate entity coverage.

References

Dhruv Agarwal, Rico Angell, Nicholas Monath, and Andrew McCallum. 2021. Entity linking and discovery via arborescence-based supervised clustering. *arXiv preprint arXiv:2109.01242*.

Rico Angell, Nicholas Monath, Sunil Mohan, Nishant Yadav, and Andrew McCallum. 2021. Clustering-based inference for biomedical entity linking. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2021)*, pages 2598–2608.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the 1998 International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference (LREC 1998)*, pages 563–566.

Samuel Broscheit. 2019. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL 2019)*, pages 677–685.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Highly parallel autoregressive entity linking with discriminative correction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7662–7669.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval. In *Proceedings of the 2021 International Conference on Learning Representations (ICLR 2021)*.

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

Greg Durrett and Dan Klein. 2014. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the Association for Computational Linguistics (TACL 2014)*, 2:477–490.

Sourav Dutta and Gerhard Weikum. 2015. C3EL: A joint model for cross-document co-reference resolution and entity linking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 846–856.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71(4):233–240.

Angela Fahrni and Michael Strube. 2012. Jointly disambiguating and clustering concepts and entities with markov logic. In *Proceedings of the 2012 International Conference on Computational Linguistics (COLING 2012)*, pages 815–832.

Hannaneh Hajishirzi, Leila Zilles, Daniel S Weld, and Luke Zettlemoyer. 2013. Joint coreference resolution and named-entity linking with multi-pass sieves. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pages 289–299.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 782–792.

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

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel S Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 5807–5812.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL 2018)*, pages 519–529.

Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 141–150.

Phong Le and Ivan Titov. 2018. Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 2018 Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 1595–1604.

Phong Le and Ivan Titov. 2019. Boosting entity linking performance by leveraging unlabeled documents. In *Proceedings of the 2019 Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 1935–1945.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on*

Empirical Methods in Natural Language Processing (EMNLP 2017), pages 188–197.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*, pages 687–692.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pages 3036–3046.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the 2005 Conference on Empirical Methods in Natural Language Processing (EMNLP 2005)*, pages 25–32.

Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng, and Michael Strube. 2014. Scoring coreference partitions of predicted mentions: A reference implementation. In *Proceedings of the 2014 Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 30–35.

William Tutte. 1984. Graph theory. *Encyclopedia of Mathematics and its Applications*, 21.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the 1995 Conference on Message understanding (MUC6, 1995)*, pages 45–52.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 2020 Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 6953–6963.

Liyan Xu and Jinho D Choi. 2020. Revealing the myth of higher-order inference in coreference resolution. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 8527–8533.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of The 2016 SIGNLL Conference on Computational Natural Language Learning (CoNLL 2016)*, pages 250–259.

Ikuya Yamada, Koki Washio, Hiroyuki Shindo, and Yuji Matsumoto. 2020. Global entity disambiguation with pretrained contextualized embeddings of words and entities. *arXiv preprint arXiv:1909.00426*.

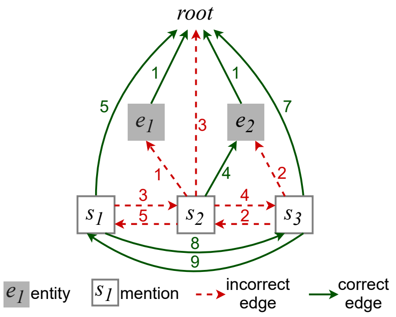


Figure 2: Illustrative graph example of Global model. The weights of the edges correspond to $\exp(\mathbf{\Phi}_{\text{cl}})$ (see eq. (11)).

Xiyuan Yang, Xiaotao Gu, Sheng Lin, Siliang Tang, Yueting Zhuang, Fei Wu, Zhigang Chen, Guoping Hu, and Xiang Ren. 2019. Learning dynamic context augmentation for global entity linking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 271–281.

Klim Zaporojets, Johannes Deleu, Chris Develder, and Thomas Demeester. 2021. DWIE: An entity-centric dataset for multi-task document-level information extraction. *Information Processing & Management*, 58(4):102563.

A Step by Step Example of MTT Theorem

In this appendix we will provide a clarifying artificial example in order to walk the reader step by step through MTT (eq. (9)–(10)) applied in our Global approach. The graph of the example is illustrated in Fig. 2 and is composed by nodes representing $root$ (r), entities e_1 and e_2, and spans s_1, s_2 and s_3. The span s_2 is associated with candidate entity set $\{e_1, e_2\}$ (i.e., represented by edges from s_2 to e_1 and e_2), and s_3 with $\{e_2\}$ (i.e., represented by the edge from s_3 to e_2). The candidate entity set of s_1 is empty. The nodes are grouped in two ground truth clusters: NIL cluster $c_1 = \{s_1, s_2\}$, and linked cluster $c_2 = \{e_2, s_2\}$.

The exponential of weighted adjacency matrix[7] $\mathbf{\Phi}_{\text{cl}}$ of the presented example is:

$$\exp(\mathbf{\Phi}_{\text{cl}}) = \begin{array}{c} \\ r \\ e_1 \\ e_2 \\ s_1 \\ s_2 \\ s_3 \end{array} \begin{array}{c} \begin{array}{cccccc} r & e_1 & e_2 & s_1 & s_2 & s_3 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 5 & 3 & 7 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 4 & 2 \\ 0 & 0 & 0 & 0 & 5 & 9 \\ 0 & 0 & 0 & 3 & 0 & 2 \\ 0 & 0 & 0 & 8 & 4 & 0 \end{bmatrix} \end{array}, \tag{11}$$

[7] For simplicity, the weights are small integers.

where the weights of incorrect edges are represented in red (i.e., red dashed edges in Fig. 2), the weights of the correct edges in green (i.e., green edges in Fig. 2), and the weights between disconnected nodes are set to 0.

In order to compute the *denominator* of the loss function in eq. (9), the Laplacian of the matrix in eq. (11) is calculated as described in eq. (10), and the row and column corresponding to root r removed (i.e., the *minor* $\mathbf{L}_r$ with respect to the root):

$$\mathbf{L}_r = \begin{array}{c} \\ e_1 \\ e_2 \\ s_1 \\ s_2 \\ s_3 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & s_1 & s_2 & s_3 \end{array} \\ \left[\begin{array}{ccccc} 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & -4 & -2 \\ 0 & 0 & 16 & -5 & -9 \\ 0 & 0 & -3 & 17 & -2 \\ 0 & 0 & -8 & -4 & 20 \end{array}\right] \end{array}. \tag{12}$$

Following Kirchhoff's Matrix Tree Theorem (Koo et al., 2007; Tutte, 1984), the determinant of $\mathbf{L}_r$ equals to the sum of the weights of all possible spanning trees of the graph represented in Fig. 2:

$$\det(\mathbf{L}_r) = 3600 = \sum_{t \in \mathcal{T}_{all}} \exp\big(\Phi_{\text{tr}}(t)\big). \tag{13}$$

In order to compute the *numerator* of the loss function in eq. (9) (i.e., the sum of the weights of the spanning trees of ground truth clusters), we first mask out (set to zero) all the weights assigned to incorrect edges:

$$\exp(\mathbf{\Phi}_{\text{cl}})' = \begin{array}{c} \\ r \\ e_1 \\ e_2 \\ s_1 \\ s_2 \\ s_3 \end{array} \begin{array}{c} \begin{array}{cccccc} r & e_1 & e_2 & s_1 & s_2 & s_3 \end{array} \\ \left[\begin{array}{cccccc} 0 & 1 & 1 & 5 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 9 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8 & 0 & 0 \end{array}\right] \end{array} \tag{14}$$

Next, the *modified Laplacian* (i.e., Laplacian with the first row replaced by root r selection weights) $\hat{\mathbf{L}}$ is calculated for both clusters c_1 and c_2:

$$\hat{\mathbf{L}}_{c_1} = \begin{array}{c} \\ r \\ s_3 \end{array} \begin{array}{c} \begin{array}{cc} s_1 & s_3 \end{array} \\ \left[\begin{array}{cc} 5 & 7 \\ -8 & 9 \end{array}\right] \end{array} \tag{15}$$

$$\hat{\mathbf{L}}_{c_2} = \begin{array}{c} \\ r \\ s_2 \end{array} \begin{array}{c} \begin{array}{cc} e_2 & s_2 \end{array} \\ \left[\begin{array}{cc} 1 & 0 \\ 0 & 4 \end{array}\right] \end{array} \tag{16}$$

The determinants of $\hat{\mathbf{L}}_{c_1}$ and $\hat{\mathbf{L}}_{c_2}$ equal to the sum of the weights of all spanning trees connecting the nodes in clusters c_1 and c_2 respectively:

$$\det(\hat{\mathbf{L}}_{c_1}) = 101 = \sum_{t \in \mathcal{T}_{c_1}} \exp\big(\Phi_{\text{tr}}(t)\big) \tag{17}$$

$$\det(\hat{\mathbf{L}}_{c_2}) = 4 = \sum_{t \in \mathcal{T}_{c_2}} \exp\big(\Phi_{\text{tr}}(t)\big) \tag{18}$$

Finally, in order to calculate the final loss, we replace the obtained results in eqs. (13), (17), and (18) in the loss function of eq. (9):

$$\mathcal{L} = -\log \frac{101 * 4}{3600}. \tag{19}$$

Note: strictly speaking, there are *three* clusters rooted in *root* in the graph of Fig. 2, the third one being $c_3 = \{e_1\}$, whose exponential weight is 1 by definition of $\Phi_{\text{cl}}(r, e_j) = 0$ (see §2.2), and has no impact in calculation of the loss function in eq. (19).

A Flexible Multi-Task Model for BERT Serving

Tianwen Wei[*] **Jianwei Qi**[*] **Shenghuan He**
Xiaomi, XiaoAI Team
{weitianwen,qijianwei,heshenghuan}@xiaomi.com

Abstract

We present an efficient BERT-based multi-task (MT) framework that is particularly suitable for iterative and incremental development of the tasks. The proposed framework is based on the idea of partial fine-tuning, i.e. only fine-tune some top layers of BERT while keep the other layers frozen. For each task, we train independently a single-task (ST) model using partial fine-tuning. Then we compress the task-specific layers in each ST model using knowledge distillation. Those compressed ST models are finally merged into one MT model so that the frozen layers of the former are shared across the tasks. We exemplify our approach on eight GLUE tasks, demonstrating that it is able to achieve 99.6% of the performance of the full fine-tuning method, while reducing up to two thirds of its overhead.

1 Introduction

In this work we explore the strategies of BERT (Devlin et al., 2019) serving for multiple tasks under the following two constraints: 1) Memory and computational resources are limited. On edge devices such as mobile phones, this is usually a hard constraint. On local GPU stations and Cloud-based servers, this constraint is not as hard but it is still desirable to reduce the computation overhead to cut the serving cost. 2) The tasks are expected to be modular and are subject to frequent updates. When one task is updated, the system should to be able to quickly adapt to the task modification such that the other tasks are not affected. This is a typical situation for applications (e.g. AI assistant) under iterative and incremental development.

In principle, there are two strategies of BERT serving: *single-task serving* and *multi-task serving*. In single-task serving, one independent single-task model is trained and deployed for each task. Typically, those models are obtained by fine-tuning a copy of the pre-trained BERT and are completely different from each other. Single-task serving has the advantage of being flexible and modular as there is no dependency between the task models. The downside is its inefficiency in terms of both memory usage and computation, as neither parameters nor computation are shared or reused across the tasks. In multi-task serving, one single multi-task model is trained and deployed for all tasks. This model is typically trained with multi-task learning (MTL) (Caruana, 1997; Ruder, 2017). Compared to its single-task counterpart, multi-task serving is much more computationally efficient and incurs much less memory usage thanks to its sharing mechanism. However, it has the disadvantage in that any modification made to one task usually affect the other tasks.

The main contribution of this work is the proposition of a framework for BERT serving that simultaneously achieves the flexibility of single-task serving and the efficiency of multi-task serving. Our method is based on the idea of partial fine-tuning, i.e. only fine-tuning some topmost layers of BERT depending on the task and keeping the remaining bottom layers frozen. The fine-tuned layers are task-specific, which can be updated on a per-task basis. The frozen layers at the bottom, which plays the role of a feature extractor, can be shared across the tasks.

2 Related Work

The standard practice of using BERT is *fine-tuning*, i.e. the entirety of the model parameters is adjusted on the training corpus of the downstream task, so that the model is adapted to that specific task (Devlin et al., 2019). There is also an alternative *feature-based* approach, used by ELMo (Peters et al., 2018). In the latter approach, the pre-trained model is regarded as a feature extractor with *frozen parameters*. During the learning of a downstream task, one feeds a fixed or learnable combination of

[*] Equal contribution.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 785 - 796
May 22-27, 2022 ©2022 Association for Computational Linguistics

L	QNLI	RTE	QQP	MNLI	SST-2	MRPC	CoLA	STS-B
1	85.9	60.3	86.1	77.1	91.6	77.2	38.7	84.8
2	88.3	63.5	88.3	80.8	91.9	80.6	40.0	86.1
3	89.9	65.3	89.0	82.5	91.2	84.6	45.3	87.3
4	90.7	69.0	89.7	83.3	92.0	84.3	48.6	88.2
5	91.0	**71.5**	90.1	84.0	92.2	**89.7**	51.3	88.3
6	91.2	71.1	90.3	84.2	93.1	86.8	53.1	86.4
7	91.3	70.0	90.5	83.9	93.0	87.5	51.5	88.6
8	91.5	70.8	90.6	84.5	92.8	88.0	**55.2**	88.9
9	91.6	70.8	90.7	84.0	92.5	87.7	54.7	88.8
10	**91.7**	69.7	**91.1**	84.5	93.0	87.3	55.0	88.7
11	**91.7**	70.4	**91.1**	84.5	93.1	88.2	54.7	**89.1**
12	91.6	69.7	**91.1**	**84.6**	**93.4**	88.2	54.7	88.8

Table 1: Dev results on GLUE datasets obtained with partial fine-tuning. The parameter L indicates the number of fine-tuned transformer layers. For each dataset and for each value of L, we always run the experiment 5 times with different initializations and report the maximum dev result obtained. The best result in each column is highlighted in bold face. Shaded numbers indicate that they attain 99% of the best result of the column. It can be seen that although fine-tuning more layers generally leads to better performance, the benefit of doing so suffers diminishing returns. Perhaps surprisingly, for RTE, MRPC and CoLA it is the partial fine-tuning with roughly half of the layers frozen that gives the best results.

the model's intermediate representations as input to the task-specific module, and only the parameters of the latter will be updated. It has been shown that the fine-tuning approach is generally superior to the feature-based approach for BERT in terms of task performance (Devlin et al., 2019; Peters et al., 2019).

A natural middle ground between these two approaches is *partial fine-tuning*, i.e. only fine-tuning some topmost layers of BERT while keeping the remaining bottom layers frozen. This approach has been studied in (Houlsby et al., 2019; Merchant et al., 2020), where the authors observed that fine-tuning only the top layers can almost achieve the performance of full fine-tuning on several GLUE tasks. The approach of partial fine-tuning essentially regards the bottom layers of BERT as a feature extractor. Freezing weights from bottom layers is a sensible idea as previous studies show that the mid layer representations produced by BERT are most transferrable, whereas the top layers representations are more task-oriented (Wang et al., 2019a; Tenney et al., 2019b,a; Liu et al., 2019a; Merchant et al., 2020). Notably, Merchant et al. (2020) showed that fine-tuning primarily affects weights from the top layers while weights from bottom layers do not alter much. Liu et al. (2019a) showed that it is possible to achieve state-of-the-art results on a number of probing tasks with linear models trained on frozen mid layer representations of BERT.

3 Method

In what follows, we denote by $\mathcal{T}$ the set of all target tasks. We always use the 12-layer uncased version of BERT as the pre-trained language model[1]. The proposed framework features a pipeline (Fig. 1) that consists of three steps: 1) Single task partial fine-tuning; 2) Single task knowledge distillation; 3) Model merging. We give details of these steps below.

3.1 Single Task Partial Fine-Tuning

In the first step, we partial fine-tune for each task an independent copy of BERT. The exact number of layers L to fine-tune is a hyper-parameter and may vary across the tasks. We propose to experiment for each task with different value of L within range $N_{\min} \leqslant L \leqslant N_{\max}$, and select the one that gives the best validation performance. The purpose of imposing the search range $[N_{\min}, N_{\max}]$ is to guarantee a minimum degree of parameter sharing. In the subsequent experiments on GLUE tasks (see Section 4.3), we set $N_{\min} = 4$ and $N_{\max} = 10$.

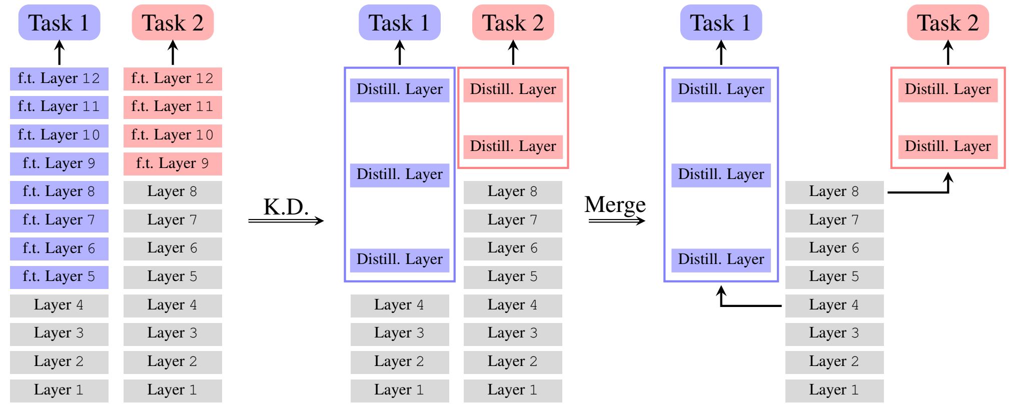
This step produces a collection of single-task models as depicted in Fig. 1(a). We shall refer to them as single-task *teacher models*, as they are to be knowledge distilled to further reduce the memory and computation overhead.

3.2 Single Task Knowledge Distillation

Since there is no interaction between the tasks, the process of knowledge distillation (KD) can be carried out separately for each task. In principle any of the existing KD methods for BERT (Wang et al., 2020; Aguilar et al., 2020; Sun et al., 2019a; Jiao et al., 2020; Xu et al., 2020a) suits our needs. In preliminary experiments we found out that as long as the student model is properly initialized, the vanilla knowledge distillation (Hinton et al., 2015) can be as performant as those more sophisticated methods.

Assume that the teacher model for task $\tau \in \mathcal{T}$ contains $L^{(\tau)}$ fine-tuned layers at the top and $12 - L^{(\tau)}$ frozen layers at the bottom. Our goal is

[1] The model checkpoint is downloaded from https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip.



(a) Teacher models (b) Student models (c) Final multi-task model

Figure 1: Pipeline of the proposed method. (a) For each task we train separately a task-specific model with partial fine-tuning, i.e. only the weights from some topmost layers (blue and red blocks) of the pre-trained model are updated while the rest are kept frozen (gray blocks). (b) We perform knowledge distillation independently for each task on the task-specific layers of the teacher models. (c) The student models are merged into one MT model so that the frozen layers of the former can be shared.

to compress the former into a smaller $l^{(\tau)}$-layer module. The proposed initialization scheme is very simple: we initialize the student model with the weights from the corresponding layers of the teacher. More precisely, let N_s denote the number of layers (including both frozen and task-specific layers) in the student, where $N_s < 12$. We propose to initialize the student from the bottommost N_s layers of the teacher. Similar approach has also been used in (Sanh et al., 2019), where the student is initialized by taking one layer out of two from the teacher. The value of $l^{(\tau)}$, i.e. the number of task-specific layers in the student model for task τ, determines the final memory and computation overhead for that task.

3.3 Model Merging

In the final step, we merge the single-task student models into one multi-task model (Fig. 1(c)) so that the parameters and computations carried out in the frozen layers can be shared. To achieve this, it suffices to load weights from multiple model checkpoints into one computation graph.

4 Experiments

In this section, we compare the performance and efficiency of our model with various baselines on eight GLUE tasks (Wang et al., 2019b). More details on these tasks can be found in Appendix A.

4.1 Metrics

The performance metrics for GLUE tasks is accuracy except for CoLA and STS-B. We use Matthews correlation for CoLA, and Pearson correlation for STS-B.

To measure the parameter and computational efficiency, we introduce the *total number of transformer layers* that are needed to perform inference for all eight tasks. For the models studied in our experiments, the actual memory usage and the computational overhead are approximately linear with respect to this number. It is named “overhead” in the header of Table 2.

4.2 Baselines

The baseline models/methods can be divided into 4 categories:

Single-task without KD. There is only one method in this category, i.e. the standard practice of single task *full fine-tuning* that creates a separate model for each task.

Single-task with KD. The methods in this category create a separate model for each task, but a certain knowledge distillation method is applied to compress each task model into a 6-layer one. The KD methods include (Hinton et al., 2015; Xu et al., 2020b; Sanh et al., 2019; Turc et al., 2019; Sun et al., 2019b; Jiao et al., 2020; Wang et al., 2020).

Multi-task learning. This category includes two versions of MT-DNN (Liu et al., 2019b, 2020), both of which produce one single multi-task model. 1) *MT-DNN (full)* is jointly trained for all eight tasks.

	QNLI	RTE	QQP	MNLI	SST-2	MRPC	CoLA	STS-B	Avg.	Layers	Overhead
Full fine-tuning	91.6	69.7	91.1	**84.6**	93.4	88.2	54.7	88.8	82.8	12 × 8	96 (100%)
DistillBERT[b]	89.2	59.9	88.5	82.2	91.3	87.5	51.3	86.9	79.6	6 × 8	48 (50.0%)
Vanilla-KD[c]	88.0	64.9	88.1	80.1	90.5	86.2	45.1	84.9	78.5	6 × 8	48 (50.0%)
PD-BERT[d]	89.0	66.7	89.1	83.0	91.1	87.2	-	-	-	6 × 8	48 (50.0%)
BERT-PKD[e]	88.4	66.5	88.4	81.3	91.3	85.7	45.5	86.2	79.2	6 × 8	48 (50.0%)
BERT-of-Theseus[f]	89.5	68.2	89.6	82.3	91.5	89.0	51.1	88.7	81.2	6 × 8	48 (50.0%)
TinyBERT[g]	90.5	72.2	90.6	83.5	91.6	88.4	42.8	-	-	6 × 8	48 (50.0%)
MiniLM[h]	88.4	66.5	88.4	81.3	91.3	85.7	45.5	86.2	79.2	6 × 8	48 (50.0%)
MT-DNN (full)[j]	91.1	**80.9**	87.6	84.4	**93.5**	87.4	51.3	86.8	82.9	12 × 1	**12** (12.5%)
MT-DNN (LOO)[k]	69.7	60.6	66.5	56.7	79.2	74.2	10.2	72.9	-	-	-
Ours (KD-1)	86.4	66.1	91.0	77.5	90.7	85.1	36.4	88.3	77.4	7 + 1×8	15 (15.6%)
Ours (KD-2)	88.6	64.6	**91.3**	81.7	92.7	86.3	44.0	88.6	79.7	7 + 2×8	23 (24.0%)
Ours (KD-3)	90.2	66.8	91.2	82.9	92.7	88.0	50.0	**88.9**	81.3	7 + 3×8	31 (32.3%)
Ours (w/o KD)	**91.7**	71.5	91.1	84.5	93.1	**89.7**	**55.2**	**88.9**	**83.2**	7 + 60	67 (69.8%)
	(2,10)	(7,5)	(2,10)	(4,8)	(6,6)	(7,5)	(4,8)	(4,8)			
Ours (mixed)	90.2	71.5	91.0	82.9	92.7	88.0	**55.2**	88.3	82.5	7 + 26	33 (34.3%)
	(2,3)	(7,5)	(2,1)	(4,3)	(6,2)	(7,3)	(4,8)	(4,1)			

Table 2: A comparison of performance and overhead between our approach and various baselines (see §4.2 for more details). The performance is evaluated on the dev set. To obtain the results labeled as "Ours", we always run the experiment 5 times with different initializations and report the maximum. The best result in each column is highlighted in bold face. Shaded numbers indicate that they attain 99% of the *Full fine-tuning* baseline. Results of [b] are from (Sanh et al., 2019); [c]-[f] are from (Xu et al., 2020b); [g]-[h] are from (Wang et al., 2020); [j]-[k] are reproduced by us with the toolkit from (Liu et al., 2020). Round bracket (x, y) indicates that the underlying task model before merging consists of x frozen layers and y task-specific layers (fine-tuned or knowledge-distilled). In the "Layers" column, notation $7 + 2 \times 8$ implies that in the final multi-task model there are 7 shared frozen layers and 2 task-specific layers for each of the 8 task.

It corresponds to the idea scenario where all tasks are known in advance. 2) *MT-DNN (LOO)*, where "LOO" stands for "leave-one-out", corresponds to the scenario where one of the eight tasks is *not* known in advance. The model is jointly pre-trained on the 7 available tasks. Then an output layer for the "unknown" task is trained with the pre-trained weights frozen.

Flexible multi-task. Our models under various efficiency constraints. *Ours (w/o KD)* means that no knowledge distillation is applied to the task models. The number of fine-tuned layers for each task is selected according to the criterion described in Section 3.1. *Ours (KD-n)* means that knowledge distillation is applied such that the student model for each task contains exactly n task-specific layers. For *Ours (mixed)*, we determine the number of task-specific layers for each task based on the marginal benefit (in terms of task performance metric) of adding more layers to the task. More precisely, for each task we keep adding task-specific layers as long as the marginal benefit of doing so is no less than a pre-determined threshold c. In Table 2, we report the result for $c = 1.0$. Results with other values of c can be found in Appendix D.

4.3 Results

The results are summarized in Table 2. From the table it can be seen that the proposed method *Ours (mixed)* outperforms all KD methods while being more efficient. Compared to the single-task full fine-tuning baseline, our method reduces up to around two thirds of the total overhead while achieves 99.6% of its performance.

We observe that MT-DNN (full) achieves the best average performance with the lowest overhead. However, its performance superiority primarily comes from one big boost on a single task (RTE) rather than consistent improvements on all tasks. In fact, we see that MT-DNN (full) suffers performance degradation on QQP and STS-B due to *task interference*, a known problem for MTL (Caruana, 1997; Bingel and Sogaard, 2017; Alonso and Plank, 2017; Wu et al., 2020). From our perspective, the biggest disadvantage of MT-DNN is that it assumes full knowledge of all target tasks in advance. From the results of MT-DNN (LOO), we observe that MT-DNN has difficulty in handling new tasks if the model is not allowed to be retrained.

5 Discussions

5.1 Advantages

One major advantage of the proposed architecture is its flexibility. First, different tasks may be fed with representations from different layers of BERT, which encapsulate different levels of linguistic information (Liu et al., 2019a). This flexibility is beneficial to both task performance and efficiency. For instance, on QQP we achieve an accuracy of 91.0, outperforming all KD baselines with *merely one* task-specific layer (connected to the 2nd layer of the frozen backbone model). Second, our architecture explicitly allows for allocating uneven resources to different tasks. We have redistributed the resources among the tasks in *ours (mixed)*, resulting in both greater performance and efficiency. Third, our framework does not compromise the modular design of the system. The model can be straightforwardly updated on on a per-task basis.

5.2 Limitations

The major limitation of our approach is that for each downstream task it requires approximately 10x more training time for the hyper-parameter search compared to the conventional approach. Although the cost is arguably manageable in practice, i.e. typically 2 or 3 days per task on a single Nvidia Tesla V100 GPU, the excessive computation load should not be overlooked.

Another limitation is that although the overall computation overhead is reduced, the *serving latency* of our model deteriorates as the number of tasks grows, and may eventually be worse than that of the single task baseline. This is due to the fact that during inference one cannot get the output of any one task until the model has finished computing for *all* tasks. In this regard, our approach may not be appropriate for those applications that demand exceptionally low serving latency, e.g. below 10 ms. Nevertheless, we report in Appendix E an industrial use case where our multi-task model serves 21 tasks while achieving a latency as low as 32 ms (99th percentile).

5.3 Comparison with Adaptor-Based Approaches

The adaptor-based approaches (Houlsby et al., 2019; Pfeiffer et al., 2020) belong to another category of fine-tuning approaches that are also parameter-efficient. Basically, the adaptor-based approaches introduce one trainable task-specific "adaptor" module for each downstream task. This module is generally lightweight, containing only a few parameters and is inserted between (or within) layers of the backbone model (e.g. BERT). However, even though the parameters of the backbone model can be shared across the tasks, the computation for inference cannot due to the fact that the internal data flow in each task model is modified by the task-specific adaptor. Therefore, the adaptor-based approaches are not computationally efficient and one needs to perform a separate full forward pass for each task. Since both parameter and computation efficiency are what we aim to achieve, the adaptor-based approaches are not comparable to our method.

6 Conclusion

We have presented our framework that is designed to provide efficient and flexible BERT-based multi-task serving. We have demonstrated on eight GLUE datasets that the proposed method achieves both strong performance and efficiency. We release our code[2] and hope that it can facilitate BERT serving in cost-sensitive applications.

References

Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. 2020. Knowledge distillation from internal representations. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7350–7357. AAAI Press.

Héctor Alonso and Barbara Plank. 2017. When is multitask learning effective? semantic sequence prediction under varying data conditions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 44–53, Valencia, Spain. Association for Computational Linguistics.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. In *TAC*.

Joachim Bingel and Anders Sogaard. 2017. Identifying beneficial task relations for multi-task learning

[2] https://github.com/DandyQi/CentraBert

in deep neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 164–169, Valencia, Spain. Association for Computational Linguistics.

Rich Caruana. 1997. Multitask Learning. *Machine Learning*, 28(1):41–75. 00000.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Z. Chen, H. Zhang, X. Zhang, and L. ZHao. 2018. Quora question pairs.

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

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019b. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.

Xiaodong Liu, Yu Wang, Jianshu Ji, Hao Cheng, Xueyun Zhu, Emmanuel Awa, Pengcheng He, Weizhu Chen, Hoifung Poon, Guihong Cao, and Jianfeng Gao. 2020. The Microsoft toolkit of multitask deep neural networks for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 118–126, Online. Association for Computational Linguistics.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 33–44, Online. Association for Computational Linguistics.

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

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy. Association for Computational Linguistics.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019a. Patient knowledge distillation for BERT model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4323–4332, Hong Kong, China. Association for Computational Linguistics.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019b. Patient knowledge distillation for BERT model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4323–4332, Hong Kong, China. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019a. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019b. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

A. Warstadt, A. Singh, and S. R. Bowman. 2018. Corpus of linguistic acceptability.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Sen Wu, Hongyang R Zhang, and Christopher Ré. 2020. Understanding and improving information transfer in multi-task learning. *arXiv preprint arXiv:2005.00944*.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020a. BERT-of-theseus: Compressing BERT by progressive module replacing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7859–7869, Online. Association for Computational Linguistics.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020b. BERT-of-theseus: Compressing BERT by progressive module replacing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7859–7869, Online. Association for Computational Linguistics.

Supplemental Materials

A Details on the GLUE tasks

The GLUE benchmark includes the following datasets:

- **QNLI** (Question Natural Language Inference). The dataset is derived from (Rajpurkar et al., 2016). This is a binary classification task where an example is of the form (question, sentence) and the goal is to predict whether the sentence contains the correct answer to the question (Wang et al., 2018).

- **RTE** (Recognizing Textual Entailment). A binary entailment task similar to MNLI but with much less training data (Bentivogli et al., 2009).

- **QQP** (Quora Question Pairs) A binary classification task where the goal is to determine if two questions asked on Quora are semantically equivalent (Chen et al., 2018).

- **MNLI** (Multi-Genre Natural Language Inference). Given a pair of sentences, the goal is to predict whether the second sentence is an entailment, contradiction or neutral with respect to the first one (Williams et al., 2018).

- **SST-2** (The Stanford Sentiment Treebank). A binary single-sentence classification task where the goal is to predict the sentiment (positive or negative) of the movie reviews (Socher et al., 2013).

- **MRPC** (Microsoft Research Paraphrase Corpus). A binary classification task where the goal is to predict whether two sentences are semantically equivalent (Dolan and Brockett, 2005).

- **CoLA** (The Corpus of Linguistic Acceptability). A binary single-sentence classification task where the goal is to predict whether an English sentence is linguistically "acceptable" or not (Warstadt et al., 2018).

- **STS-B** (The Semantic Textual Similarity Benchmark). A regression task where the goal is to predict whether two sentences are similar in terms of semantic meaning as measured by a score from 1 to 5 (Cer et al., 2017).

- **WNLI** (Winograd NLI). The dataset is derived from (Levesque et al., 2012). We exclude this task in our experiments following the practice of (Devlin et al., 2019; Radford et al., 2018).

Dataset	Train	Dev
QNLI	108k	5.4k
RTE	2.5k	0.3k
QQP	363k	40k
MNLI	392k	9.8k
SST-2	67k	0.8k
MRPC	3.5k	0.4k
CoLA	8.5k	1.0k
STS-B	5.7k	1.5k

Table 3: Number of examples for training and development in GLUE datasets.

B Hyper-parameters

The approach presented in this work introduces two new hyper-parameters for each task $\tau \in \mathcal{T}$, namely the number of fine-tuned layers $L^{(\tau)}$ for the teacher and the number of knowledge distilled layer $l^{(\tau)}$ for the student. If the resources permit, these two hyper-parameters should be tuned separately for each task. As introduced in Section 3.1, we suggest to constrain L within the range $4 \leq L^{(\tau)} \leq 10$. As for $l^{(\tau)}$ which determines the eventual task-specific overhead, we impose $l^{(\tau)} \leq 3$. Since we always determines $L^{(\tau)}$ first, we do not need to experiment with every combination of $(L^{(\tau)}, l^{(\tau)})$. Combining these together, our approach requires approximately 10x (7 for L and 3 for l) more training time compared to conventional full fine-tuning approach.

The conventional hyper-parameters (e.g. learning rate, mini-batch size, etc) used in our experiments are summarized in Table 4.

C Detailed Experiment Results

In the box plots of Figure 2 above we report the performance of the student models initialized from pre-trained BERT and from the teacher. It can be clearly seen that the latter initialization scheme generally outperforms the former. Besides, we also observe that although increasing the number of task-specific layers improves the performance, the marginal benefit of doing so varies across tasks.

Hyper-parameter	value
learning rate	2e-5
batch size	32
Epoch	3, 4, 5
Optimizer	Adam
weight decay rate	0.01
β_1	0.9
β_2	0.999
ϵ	1e-6

Table 4: Hyper-parameters used in our experiments. We mainly followed the practice of (Devlin et al., 2019).

Notably, for QQP and STS-B the student models with only one task-specific layer are able to attain 99% of the performance of their teacher.

D Performance-Efficiency Trade-off

In Fig 5, we report the performance of our method with various values of c, where c is defined as the minimal marginal benefit (in terms of task performance metric) that every task-specific layer should bring (see Section 4.2).

E Industrial Application

We have implemented our framework in the application of utterance understanding of XiaoAI, a mono-lingual (Chinese) commercial AI assistant developed by XiaoMi. Our flexible multi-task model forms the bulk of the utterance understanding system, which processes over 100 million user queries per day with a peak throughput of nearly 4000 queries-per-second (QPS).

For each user query, the utterance understanding system performs various tasks, including emotion recognition, incoherence detection, domain classification, intent classification, named entity recognition, slot filling, etc. Due to the large workload, these tasks are developed and maintained by a number of different teams. As the AI assistant itself is under iterative/incremental development, its utterance understanding system undergoes frequent updates[3]:

- Update of training corpus, e.g. when new training samples become available or some mislabeled samples are corrected or removed.
- Redefinition of existing tasks. For instance, when a more fine-grained intent classification is needed, we may need to redefine existing intent labels or introduce new labels.
- Introduction of new tasks. This may happen when the AI assistant needs to upgrade its skillsets so as to perform new tasks (e.g. recognize new set of instructions, play verbal games with kids, etc).
- Removal of obsolete tasks. Sometimes a task is superseded by another task, or simply deprecated due to commercial considerations. Those tasks need to be removed from the system.

One imperative feature for the system is the *modular design*, i.e. the tasks should be independent of each other so that any modification made to one task does not affect the other tasks. Clearly, a conventional multi-task system does not meet our need as multi-task training breaks modularity.

Before the introduction of BERT, our utterance understanding system is based on single-task serving, i.e. a separate model is deployed for each task. As those models are relatively lightweight (e.g. TextCNN, LSTM), overhead is not an issue. However, with the introduction of BERT, the cost for single-task serving becomes a valid concern as each task model (a unique 12-layer fine-tuned BERT) requires two Nvidia Tesla V100 GPUs for stable serving that meets the latency requirement.

With the primary objective of reducing cost, we have implemented the proposed flexible multi-task model in our utterance understanding system, which provides serving for a total of 21 downstream tasks. Overall, there are 40 transformer layers of which 8 are shared frozen layers (on average 1.5 task-specific layers per task). Using only 5 Nvidia Tesla V100 GPUs, we are able to achieve[4] a P99 latency of 32 ms under a peak throughput of 4000 QPS. Compared with single-task serving for 21 tasks which would require 42 GPUs, we estimate that our system reduces the total serving cost by up to 88%.

[3] Not necessarily frequent for any particular task, but overall frequent if we regard the system as a whole.

[4] with fp16 and fast transformer (`https://github.com/NVIDIA/FasterTransformer`) acceleration.

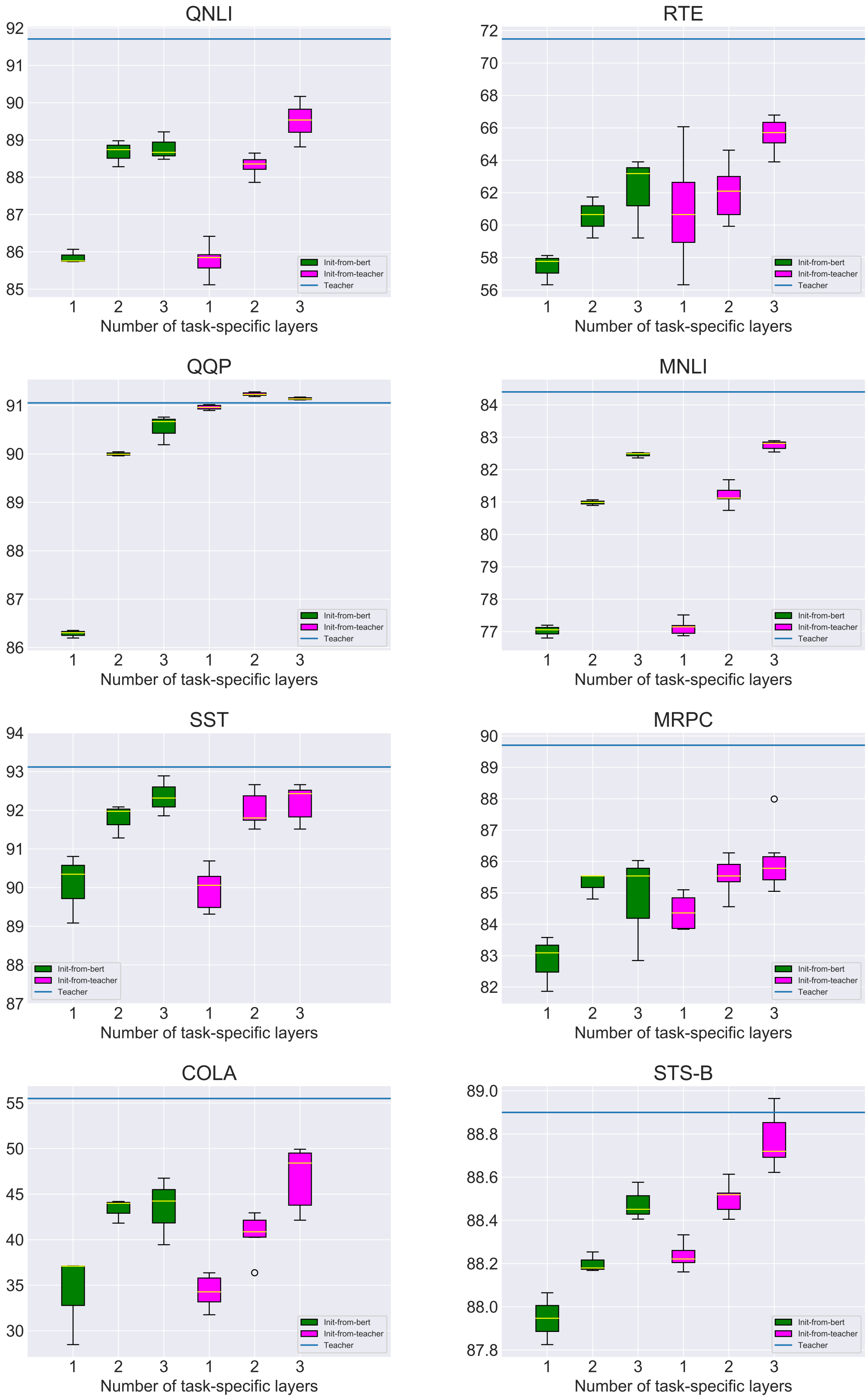


Figure 2: A comparison of the task performance between vanilla initialization (initialize from pre-trained BERT) and teacher initialization as described in Section 3.2 for $n \in \{1, 2, 3\}$, where n is the number of task-specific layers in the student model.

	QNLI	RTE	QQP	MNLI	SST-2	MRPC	CoLA	STS-B	Avg.	Layers	Overhead
Full fine-tuning	91.6	69.7	91.1	**84.6**	93.4	88.2	54.7	88.8	82.8	12 × 8	96 (100%)
Ours (KD-1)	86.4	66.1	91.0	77.5	90.7	85.1	36.4	88.3	77.4	7 + 1×8	15 (15.6%)
Ours (KD-2)	88.6	64.6	**91.3**	81.7	92.7	86.3	44.0	88.6	79.7	7 + 2×8	23 (24.0%)
Ours (KD-3)	90.2	66.8	91.2	82.9	92.7	88.0	50.0	**88.9**	81.3	7 + 3×8	31 (32.3%)
Ours ($c = 1.0$)	90.2	71.5	91.0	82.9	92.7	88.0	**55.2**	88.3	82.5	7 + 26	33 (34.3%)
	(2,3)	(7,5)	(2,1)	(4,3)	(6,2)	(7,3)	(4,8)	(4,1)			
Ours ($c = 2.0$)	88.6	66.1	91.0	81.7	92.7	85.1	50.0	88.3	80.4	7 + 13	20 (20.2%)
	(2,2)	(7,1)	(2,1)	(4,2)	(6,2)	(7,1)	(4,3)	(4,1)			
Ours ($c = 3.0$)	86.4	66.1	91.0	81.7	90.7	85.1	50.0	88.3	79.9	7 + 11	18 (18.8%)
	(2,1)	(7,1)	(2,1)	(4,2)	(6,1)	(7,1)	(4,3)	(4,1)			
Ours (w/o KD)	**91.7**	71.5	91.1	84.5	93.1	**89.7**	**55.2**	**88.9**	**83.2**	7 + 60	67 (69.8%)
	(2,10)	(7,5)	(2,10)	(4,8)	(6,6)	(7,5)	(4,8)	(4,8)			

Table 5: Results with various values of c. This parameter controls the performance-efficiency trade-off of the overall multi-task model, in the sense that we allow the growth of an existing task module by one more task-specific layer only if that would bring a performance gain greater than c.

Responsible NLP Research Checklist

A. For every submission

A1. Did you discuss the *limitations* of your work?
Yes, it is explicitly discussed in Section 5.2.

A2. Did you discuss any potential risks of your work?
No, we believe that there is no potential risk.

A3. Do the abstract and introduction summarize the paper's main claims?
Yes, we confirm so.

B. Did you use or create scientific artifacts?

Yes, we used the GLUE datasets in Section 4.

B1. Did you cite the creators of artifacts you used?
Yes, the GLUE paper is cited in Section 4. The individual datasets in GLUE are cited in Appendix A.

B2. Did you discuss the license or terms for use and/or distribution of any artifacts?
No. Since those artifacts are popular in the NLP community, we merely followed the common practice of using these artifacts. We do not believe that our usage violate the license for use, or is potentially risky in any ways we can imagine.

B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No. The justification is the same that for question B2.

B4. Did you discuss the steps taken to check whether the data that was collected/used con- tains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No. The justification is the same that for question B2.

B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Yes, it is provided in Appendix A.

B6. Did you report relevant statistics like the number of examples, details of train/test/dev splits, etc. for the data that you used/created?
Yes, it is provided in Appendix A.

C. Did you run computational experiments?

Yes, in Section 4.

C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure

used?
No, we did not report the number of parameters in the models used as it can be easily inferred from Table 2. The total computation budget was discussed in Section 5.2.

C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Yes, it is provided in the Section 4 and Appendix B.

C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Yes, we explicitly stated in the caption of Table 1 and Table 2 that our results are the maximum over 5 independent runs. Detailed results are also reported in Appendix C.

C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
We did reuse the WordPiece implementation from BERT's repository `https://github.com/google-research/bert` *for tokenization. We did not report this as we consider it as a trivial matter.*

D. Did you use human annotators (e.g., crowdworkers) or research with human subjects?

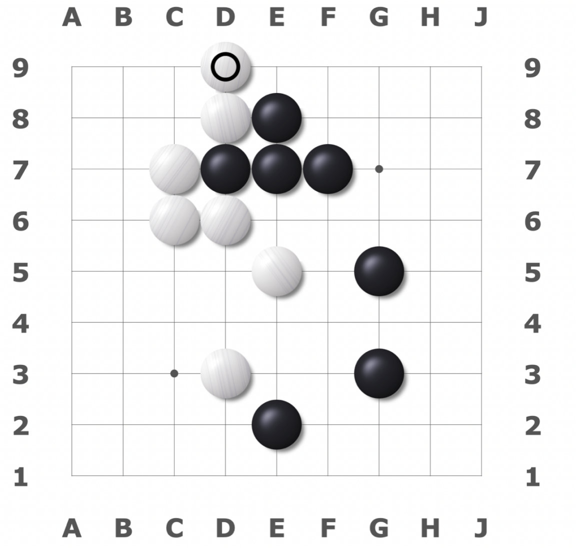
No, we did not use any human annotators, nor did we research with human subjects.

Understanding Game-Playing Agents with Natural Language Annotations

Nicholas Tomlin **Andre He** **Dan Klein**
Computer Science Division, University of California, Berkeley
{nicholas_tomlin, andre.he, klein}@berkeley.edu

Abstract

We present a new dataset containing 10K human-annotated games of Go and show how these natural language annotations can be used as a tool for model interpretability. Given a board state and its associated comment, our approach uses linear probing to predict mentions of domain-specific terms (e.g., *ko*, *atari*) from the intermediate state representations of game-playing agents like AlphaGo Zero. We find these game concepts are nontrivially encoded in two distinct policy networks, one trained via imitation learning and another trained via reinforcement learning. Furthermore, mentions of domain-specific terms are most easily predicted from the later layers of both models, suggesting that these policy networks encode high-level abstractions similar to those used in the natural language annotations.



“Bad **shape**. If white wants to defend it should be solid at c8, leaving no weaknesses or **sente** moves for black.”

Figure 1: Example comment from our dataset, with domain-specific keywords (*shape*, *sente*) highlighted. Although this comment is from a 9 × 9 game for illustrative purposes, our dataset primarily focuses on annotations from 19 × 19 games.

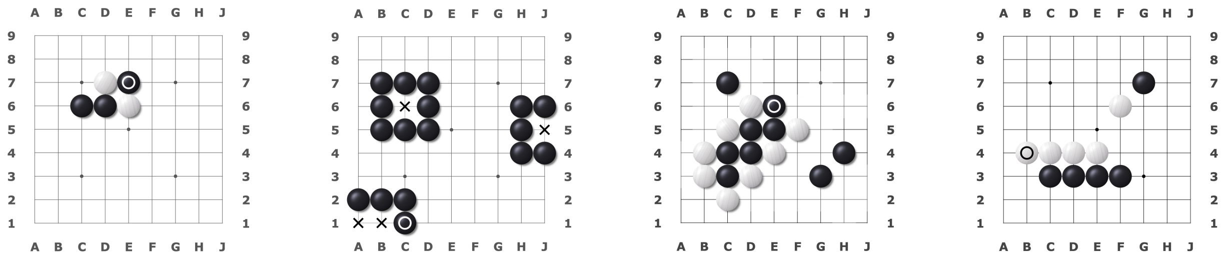
1 Introduction

Go is fundamentally a game of pattern recognition: from *ladders* and *walls* to *sente* and *shape*, professional players rely on a rich set of concepts to communicate about structures on the game board. Some patterns are relatively simple: *walls* are lines of adjacent stones, and an *atari* is a threat to capture stones on the next move; other patterns are less clearly defined: *hane* refers to any move that “goes around” the opponent’s stones, and *sente* describes a general state of influence or tempo. Despite the nebulous definitions of some of these terms, human players use them productively. Beginners learn about *eyes* that determine when groups of stones are *alive* or *dead* and are given guidelines for when they should play a *cut* or extend a *ladder*; more advanced players learn sequences of *joseki* and *tesuji* and are taught to distinguish *good shape* from *bad shape*. Figures 1-2 depict some example concepts.

Computers have recently surpassed human performance at Go (Silver et al., 2016), but relatively little is known about why these programs perform so well and whether they rely on similar representational units to choose the moves they play. While post-hoc behavioral analyses suggest that AlphaGo and its successor AlphaGo Zero (Silver et al., 2017) can process complex game situations involving *shape*, *capturing races*, *sente*, *tesuji*, and even *ladders*, existing interpretability work has focused on the moves that agents play, rather than the internal computations responsible for those moves. Our work instead proposes a *structural* analysis by correlating the internal representations of game-playing agents with information from a naturally-occurring dataset of move-by-move annotations.

In this paper, we use linear probing to explore how domain-specific concepts are represented by

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 797 - 807
May 22-27, 2022 ©2022 Association for Computational Linguistics



(a) Cut (b) Eyes (c) Ladder (d) Wall

Figure 2: Example Go patterns and associated terminology. (a) *Cuts* are moves that separate two groups of stones. (b) *Eyes* are empty squares surrounded by stones of the same color. (c) *Ladders* are capturing races which may span the entire board. (d) Walls are lines of adjacent stones of the same color. These terms appear frequently in our dataset of natural language annotations and are further defined in the appendix.

game-playing agents. Because we do not have ground-truth labels explaining which concepts are relevant to a given game state, we collect a dataset of 10K annotated Go games (§2.1). Given a board state and its associated comment, we produce binary feature vectors summarizing which game phenomena (e.g., *ko*, *atari*) are mentioned in the comment and use pattern-based feature extractors to determine which phenomena are actually present on the board (§2.2). We then feed board states into two policy networks with disparate architectures and training methods (§3.1) to obtain intermediate representations. Finally, we use linear probes (§3.2) to predict the binary feature vectors from our policy networks. Generally, we find that pattern-based features are encoded in the early layers of policy networks, while natural language features are most easily extracted from the later layers of both models. We release our code and data at `https://github.com/andrehe02/go`.

2 Dataset

2.1 Annotated Games

We collect 10K games with move-by-move English annotations from the Go Teaching Ladder (GTL).[1] The GTL was created by Jean-loup Gailly and Bill Hosken in 1994 and maintained until 2016 and permits non-commercial digital redistribution. Until 2016, members of the GTL could submit games for review by volunteers, who ranged from amateur to professional strength. Reviewers were given annotation guidelines and required to have a higher rating than their assigned reviewees, resulting in high quality natural language data. Of the collected games, we focus on 9524 which were played on classical 19 × 19 boards; many games also include unplayed analysis variations which we do not use in this work. These 9524 games contain 458,182 total comments, with a median length of 14 words.

2.2 Feature Extraction

We convert board states and comments into binary feature vectors using two methods: (1) *pattern-based* feature extraction, which checks for the ground truth presence of features from the board state, and (2) *keyword-based* feature extraction, which converts comments into bag-of-words representations based on domain-specific keywords.

Pattern-Based We define a set of rules to determine which game phenomena are present in a given board state, including: *cuts*, *eyes*, *ladders*, and *walls*. For example, we decide that a *wall* is present when four stones of the same color are placed in a row adjacent to one another. Because patterns like *wall* and *cut* are often imprecisely defined, these definitions may not align perfectly with player intuitions; we therefore provide additional details for each phenomena in the appendix. We do not attempt to write rule-based definitions of vaguer concepts like *sente* and *influence*.

Keyword-Based We scrape an online vocabulary of domain-specific terminology[2] and find the 30 most common terms in our natural language annotations. We then convert each comment into a 30-dimensional binary feature vector representing whether or not it contains these keywords; we additionally include features based on 60 control words, chosen according to frequency statistics, which are further subdivided into function and content words. Our wordlist and details about our selection of control words can be found in the appendix.

[1] `https://gtl.xmp.net`

[2] `https://senseis.xmp.net/?GoTerms`

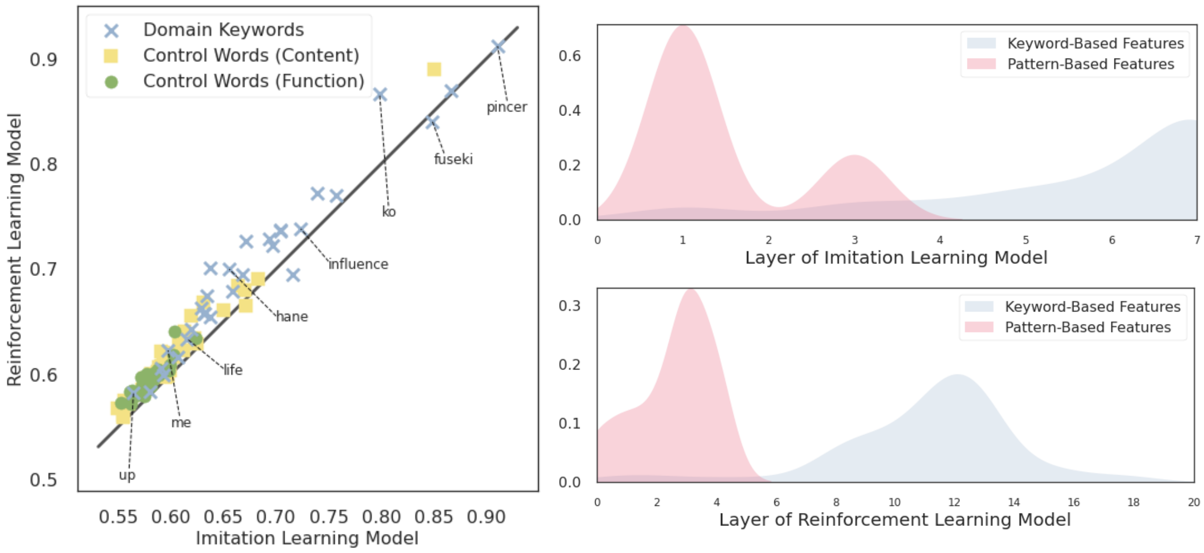


Figure 3: Results for the imitation learning and reinforcement learning agents are highly correlated. (Left) Scatterplot of ROC AUC values for linear probes trained to predict the presence of domain-specific keywords in move-by-move annotations. Keywords but not control words are predictable from the intermediate layers of both models. Words to the left of the solid line ($y = x$) are better predicted from the reinforcement learning model. (Right) Kernel density estimates showing where information is best represented in the policy networks (cf. §3.3). For both policy networks, pattern-based features are encoded in early layers, while keyword-based features are most easily extracted from later layers. Layer 0 denotes the input board representation for both models.

Having both pattern-based and keyword-based features captures a trade-off between precision and coverage. Writing rules for pattern-based features is labor-intensive and essentially impossible for many game concepts. Meanwhile, keyword-based features are inherently noisy: comments often mention phenomena which didn't actually occur in the game, and common structures like *atari* and *eyes* are frequently left unmentioned because the annotator and players already know they exist. Nonetheless, we find that probes are capable of predicting the presence of domain-specific keywords with significantly better-than-chance accuracy.

3 Methods

3.1 Policy Networks

We analyze two agents: (1) an imitation learning agent using the architecture described in Clark and Storkey (2015), and (2) a pre-trained ELF OpenGo model (Tian et al., 2017, 2019), which is an open-source, reinforcement learning agent similar to AlphaGo Zero (Silver et al., 2017). Our imitation learning model was trained on 228,000 games and achieved a rating of 1K (≈ 1900 ELO) on the Online Go Server (OGS),[3] where it played against a combination of humans and computers until its rating stabilized. ELF OpenGo reports a self-play ELO over 5000, but this metric is inflated (Tian et al., 2019). Although we refer to these agents by their training procedure (i.e., imitation vs. reinforcement), there are several other differences between the models. One possible source of variance between agents involves the format of the board state representation. Following Clark and Storkey (2015), our imitation learning model takes as input a $19 \times 19 \times 7$ binary matrix. Of the seven planes, six represent the positions of stones, divided by color and the number of *liberties*; the seventh plane represents *ko* information. Meanwhile, the reinforcement learning model's $19 \times 19 \times 17$ input contains a partial history of the game state.

3.2 Linear Probes

Given a board state and paired feature vector as described in Section 2.2, we compute intermediate representations by feeding the board state into frozen policy networks. To predict each feature of interest, we run logistic regression independently on each layer of each policy network, including the raw board state. In other words, for each policy network, we train $F \times L \times k$ classifiers, where F is the number of features, L is the number of layers, and k is the parameter for k-fold cross-validation, as discussed in the following section.

[3] https://online-go.com

Domain Word	Imitation	Reinforcement	Rough Definition
Pincer	0.91	0.91	attack on a corner approach
Joseki	0.87	0.87	fixed local sequences of moves
Fuseki	0.85	0.84	opening
Ko	0.80	0.86	repetitive capture sequence
Wall	0.70	0.74	sequence of stones in a row
Atari	0.69	0.73	threat to capture
Eye	0.67	0.73	surrounded empty space
Cut	0.64	0.65	block two groups from connecting
Me	0.60	0.62	another word for *eye*
Down	0.60	0.60	toward the edge of the board
Point	0.59	0.61	specific locations on the board; or, the score
Force	0.58	0.58	requiring immediate response
Up	0.56	0.58	toward the center of the board

Table 1: ROC AUC values for a subset of domain words in both the imitation learning and reinforcement learning models. Higher values correspond to more predictable words. Domain words with the highest values represent relatively straightforward corner patterns (*pincer*), while keywords with the lowest values (*force*, *up*) are polysemous with commonly used non-domain-specific meanings. See Table 2 in the appendix for additional ROC AUC values.

3.3 Metrics

We seek to answer two questions: (1) *what* information is represented in the policy networks, and (2) *where* is this information represented? To answer the first question, we compute the area under the receiver operating characteristic curve (ROC AUC) for each linear probe. Specifically, for each layer, we compute the average ROC AUC after 10-fold cross-validation and then take the maximum average value across layers. Features with high ROC AUC are said to be *represented* by a model, because they are linearly extractible from some intermediate layer of its policy network. To answer the second question, we compute the layer at which each feature has its highest ROC AUC value; we then apply 10-fold cross-validation, summarize the counts for each feature in a histogram, and compute a kernel density estimate (KDE) for visualization.

4 Results

We find that domain-specific keywords are significantly more predictable than control words, with $p = 1.8 \times 10^{-5}$ under the Wilcoxon signed-rank test. As shown in Figure 3 (Left) and Table 1, the keyword with the highest ROC AUC value across both models is *pincer*, which denotes a relatively straightforward corner pattern. Meanwhile, low-valued domain words like *me* and *up* are polysemous with non-domain-specific meanings and therefore difficult to predict. While content and function control words have roughly similar distributions, some content words are noticeably more predictable; for example, *opponents* is the highest-valued control word with ROC AUC values of $(0.85, 0.89)$ as seen in Figure 3 (Left). Such control words are likely predictable due to correlations with certain domain-specific concepts.

ROC AUC values for the two models are strongly correlated, with Pearson's coefficient $\rho = 0.97$. Figure 3 (Left) shows that for most keywords, the reinforcement learning model slightly outperforms the imitation learning model. Furthermore, keywords are significantly more predictable from the imitation learning model than from a randomly initialized baseline with identical architecture ($p = 5.6 \times 10^{-16}$). Some words like *ko* are noticeably more predictable from the reinforcement learning model, possibly due to differences in input board state representations (cf. §3.1); further discussion of this point can be found in the appendix.

Consistent with our knowledge that pattern-based features can be obtained by applying simple rules to the raw board state, we find that pattern-based features are encoded in early layers of both models, as shown in Figure 3 (Right). Meanwhile, keyword-based features are most easily extracted from later layers, suggesting that they correlate with high-level abstractions in the policy network. Generally, pattern-based features are much more predictable than keyword-based features, with average ROC AUC values of $(0.96, 0.98)$ and

$(0.68, 0.70)$, respectively. As discussed in Section 2.2, this discrepancy can largely be attributed to the noisiness inherent in natural language data.

5 Related Work

Jhamtani et al. (2018) propose a similarly-sized dataset of move-by-move chess commentary. Rather than using this commentary for model interpretability, though, Jhamtani et al. (2018) attempt to predict whole comments from raw board states. Zang et al. (2019) use the same dataset to jointly train a policy network and language generation model with a shared neural encoder, but again focus on the pedagogical application of commentary generation rather than interpretation of the policy network. Similar work has focused on generating sportscasts in the Robocup domain (Chen and Mooney, 2008; Liang et al., 2009; Mei et al., 2016).

Our primary methodology is linear probing (Ettinger et al., 2016; Manning et al., 2020), which has commonly been used to study the intermediate representations of language models like ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019). One classic result in this area shows that early layers of contextual language models correlate best with lexical-syntactic information such as part of speech, while later layers correlate with semantic information like proto-roles and coreference (Tenney et al., 2019). Recent work on control tasks (Hewitt and Liang, 2019), minimum description length (Voita and Titov, 2020), and Pareto probing (Pimentel et al., 2020) has focused on improving the methodological rigor of this paradigm. Although linear probing is fundamentally a correlational method, other recent work has focused on whether information which is easily extractable from intermediate layers of a deep network is causally used during inference (Elazar et al., 2021; Lovering et al., 2021).

Most related to our work are contemporary studies by McGrath et al. (2021) and Forde et al. (2022), which apply probing techniques to the games of chess and Hex, respectively. McGrath et al. (2021) use linear probes to predict a large number of pattern-based features throughout the training of an AlphaZero agent for chess. Meanwhile, Forde et al. (2022) train linear probes for pattern-based features on an AlphaZero agent for Hex and run behavioral tests to measure whether the agent "understands" these concepts. Comparatively, our work uses fewer features than McGrath et al. (2021) and does not make causal claims about how representations are used during inference, as in Forde et al. (2022); however, to the best of our knowledge, our work is the first of its kind to use features derived from natural language in conjunction with probing techniques for policy interpretability.

6 Conclusion

We presented a new dataset of move-by-move annotations for the game of Go and showed how it can be used to interpret game-playing agents via linear probes. We observed large differences in the predictability of pattern-based features, which are extracted from the board state, and keyword-based features, which are extracted from comments. In particular, pattern-based features were easily extracted from lower layers of the policy networks we studied, while keyword-based features were most predictable from later layers. At a high level, this finding reinforces the intuition that written annotations describe high-level, abstract patterns that cannot easily be described by a rule-based approach. Accordingly, we argue there is much to learn from this annotation data: future work might attempt to correlate policy network representations with richer representations of language, such as those provided by a large language model. Future work might also explore whether similar approaches could be used to improve game-playing agents, either by exposing their weaknesses or providing an auxiliary training signal. We also expect similar approaches may be viable in other reinforcement learning domains with existing natural language data.

Acknowledgements

We are grateful to Kevin Yang for his work on the imitation learning model, and to Rodolfo Corona and Ruiqi Zhong for their contributions to an early version of this project. We thank Roma Patel, the members of the Berkeley NLP Group, and our anonymous reviewers for helpful suggestions and feedback. This work was supported by the DARPA XAI and LwLL programs and a National Science Foundation Graduate Research Fellowship.

References

David L Chen and Raymond J Mooney. 2008. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135.

Christopher Clark and Amos Storkey. 2015. Training deep convolutional neural networks to play Go. In

International conference on machine learning, pages 1766–1774. PMLR.

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

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2021. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175.

Allyson Ettinger, Ahmed Elgohary, and Philip Resnik. 2016. Probing for semantic evidence of composition by means of simple classification tasks. In *Proceedings of the 1st workshop on evaluating vector-space representations for nlp*, pages 134–139.

Jessica Zosa Forde, Charles Lovering, George Konidaris, Ellie Pavlick, and Michael L. Littman. 2022. Where, when which concepts does AlphaZero learn? lessons from the game of Hex. In *AAAI Workshop on Reinforcement Learning in Games*.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743.

Harsh Jhamtani, Varun Gangal, Eduard Hovy, Graham Neubig, and Taylor Berg-Kirkpatrick. 2018. Learning to generate move-by-move commentary for chess games from large-scale social forum data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1661–1671.

Percy Liang, Michael I Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99.

Charles Lovering, Rohan Jha, Tal Linzen, and Ellie Pavlick. 2021. Predicting inductive biases of pretrained models. In *International Conference on Learning Representations*.

Christopher D Manning, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences*, 117(48):30046–30054.

Thomas McGrath, Andrei Kapishnikov, Nenad Tomašev, Adam Pearce, Demis Hassabis, Been Kim, Ulrich Paquet, and Vladimir Kramnik. 2021. Acquisition of chess knowledge in alphazero. *arXiv preprint arXiv:2111.09259*.

Hongyuan Mei, TTI UChicago, Mohit Bansal, and Matthew R Walter. 2016. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. In *Proceedings of NAACL-HLT*, pages 720–730.

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

Tiago Pimentel, Naomi Saphra, Adina Williams, and Ryan Cotterell. 2020. Pareto probing: Trading off accuracy for complexity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3138–3153. Association for Computational Linguistics.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.

Yuandong Tian, Qucheng Gong, Wenling Shang, Yuxin Wu, and C. Lawrence Zitnick. 2017. Elf: An extensive, lightweight and flexible research platform for real-time strategy games. In *Advances in Neural Information Processing Systems*, pages 2656–2666.

Yuandong Tian, Jerry Ma, Qucheng Gong, Shubho Sengupta, Zhuoyuan Chen, James Pinkerton, and Larry Zitnick. 2019. ELF OpenGo: An analysis and open reimplementation of alphazero. In *International Conference on Machine Learning*, pages 6244–6253. PMLR.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196.

Hongyu Zang, Zhiwei Yu, and Xiaojun Wan. 2019. Automated chess commentator powered by neural chess engine. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5952–5961.

A Dataset Statistics

The most common domain-specific terms appear in more than 15K comments, as shown in Figure 4.

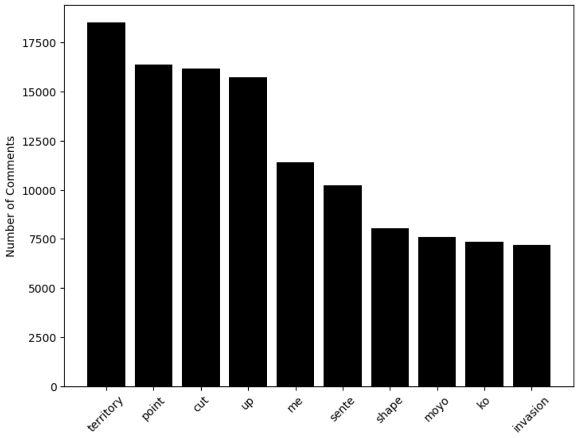


Figure 4: Histogram showing the number of comments that contain each of the ten most common domain-specific words.

B Pattern-Based Feature Extraction

As described in Section 2.2, we extract features from the board state using a set of hand-crafted rules. These rules may not align perfectly with player intuitions, which can be hard to formulate concisely, and so are presented here for full detail.

Cut We define *cuts* as moves that prevent the opponent from connecting two disconnected groups on their next move. To avoid labelling squares where a play would be immediately capturable, we also require that a cut have at least two liberties. Note that this definition permits non-diagonal cuts.

Eye We define *eyes* as connected groups of empty squares that are completely surrounded by stones of the same color. We require there be no enemy stones in the same surrounded region, so this definition fails to capture eyes that surround *dead* stones.

Ladder The term *ladder* describes the formation shown in Figure 2c. Since human players can usually predict who wins a ladder, they rarely play out the capturing race. For this reason, we do not look for ladder formations, but instead label moves that would start or continue a ladder. Specifically, we label a square for the ladder feature if it is the singular liberty of a friendly group of stones and a play at the square results in the group having exactly two liberties. We do not count trivial ladders that lie at the edge of the board.

Wall We define a *wall* as a connected row or column with four or more stones of the same color.

C Keywords and Control Words

Keywords We choose the first thirty most frequent terms (cf. Table 1) from our vocabulary of domain-specific terminology as keywords: *territory, point, cut, sente, up, me, moyo, shape, ko, invasion, influence, wall, joseki, eye, alive, gote, life, pincer, aji, thickness, base, atari, connected, hane, tenuki, down, overplay, force, reading, fuseki*.

Control Words Our control words consist of the thirty most frequent words in our dataset, as well as thirty words uniformly distributed according to the same frequency as the keywords: *the, is, to, this, white, black, and, you, at, for, in, move, it, of, but, not, be, have, play, that, on, good, here, if, better, can, would, now, should, stones, looking, wanted, opponents, wasnt, defending, save, youre, answer, three, fine, feel, place, lose, bit, possibility, attacking, likely, leaves, shouldnt, question, lost, threat, almost, theres, continue, trying, hope, just, exchange, before*. We further subdivide the control words based on whether or not they appear in the NLTK stopword list,[4] which we use as a rough proxy for distinguishing between function and content words.

D Additional Results

We additionally report de-aggregated ROC AUC values for each keyword across layers, as shown in Figures 5-7. These figures show the raw data used to compute the kernel density estimates in Figure 3, which show that natural language features are most easily extracted from later layers of both models. We note in Figure 5 that *ladders* are the most difficult pattern-based feature to predict, which is consistent with our knowledge that many Go-playing agents fail to correctly handle ladders without special feature engineering (Tian et al., 2019). Anecdotally, our imitation learning model often failed to play ladders correctly; this is consistent with the finding that ladders are more predictable from the reinforcement learning model. Future work might investigate whether this probing framework could be used to effectively predict model behavior in situations like these, as in Forde et al. (2022) for the game of Hex.

[4] https://gist.github.com/sebleier/554280

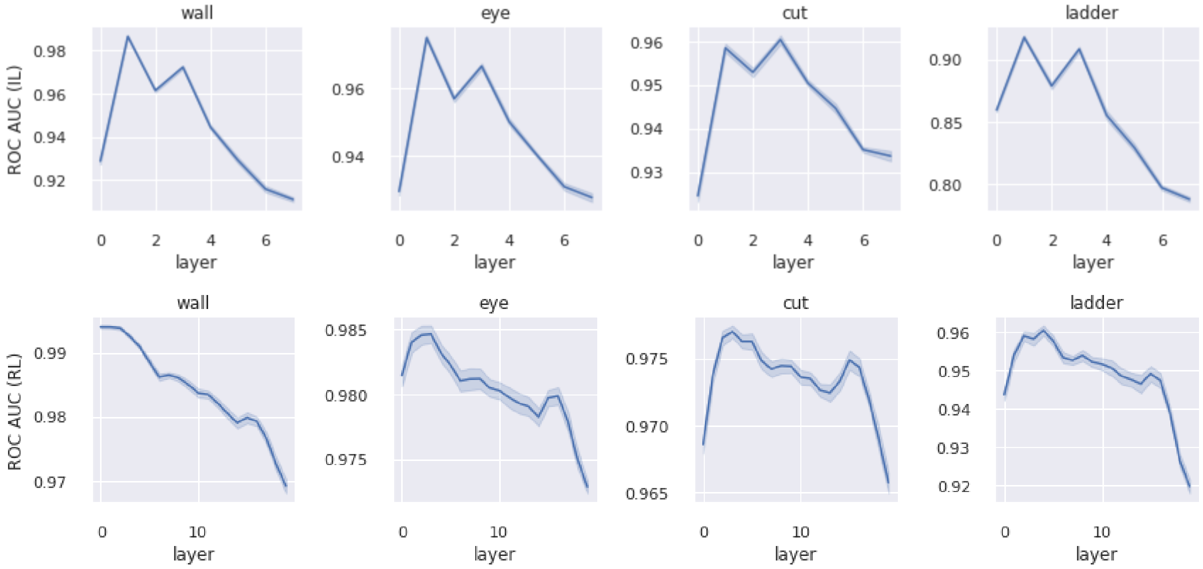


Figure 5: Plots of ROC AUC values for pattern-based features in the imitation learning model (top) and the reinforcment learning model (bottom). Among the four pattern-based features we consider, *ladders* have by far the lowest ROC AUC values. As noted in Tian et al. (2019), *ladders* are a known challenge for Go agents, requiring special feature engineering in Silver et al. (2016). Therefore, perhaps it is unsurprising that *ladders* were the most difficult pattern-based feature to predict.

E Major Differences Between Imitation and Reinforcement Learning Models

While most keywords have similar ROC AUC values across models, *ko*, *eye*, *atari*, and *overplay* have a noticeably higher ROC AUC values under the reinforcement learning model (cf. Table 1). However, this discrepancy is not obviously attributable to the difference in training procedures (i.e., imitation vs. reinforcement). As described in Section 3.1, the two models use different input state representations, which differ in their encoding of *ko* and *liberty* information, which is used to determine whether *eyes* and *atari* exist. Such architectural differences may explain discrepancies across models, but do not account for words like *overplay*; playing strength is another possible (but not confirmed) source of these discrepancies.

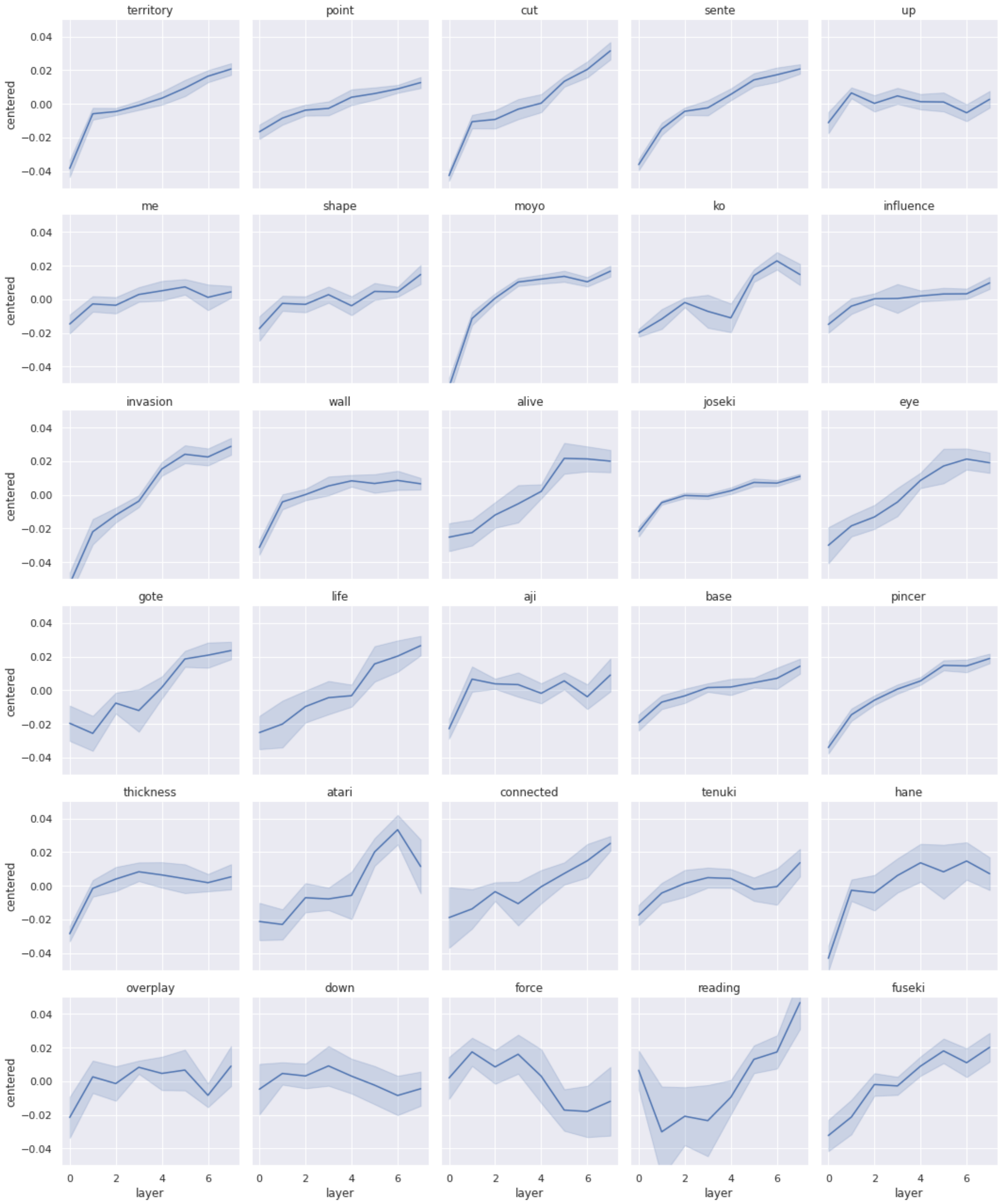


Figure 6: Plots of centered ROC AUC values for keywords in the imitation learning model. For most keywords, linear probe performance peaks at mid-to-late layers of the imitation learning model.

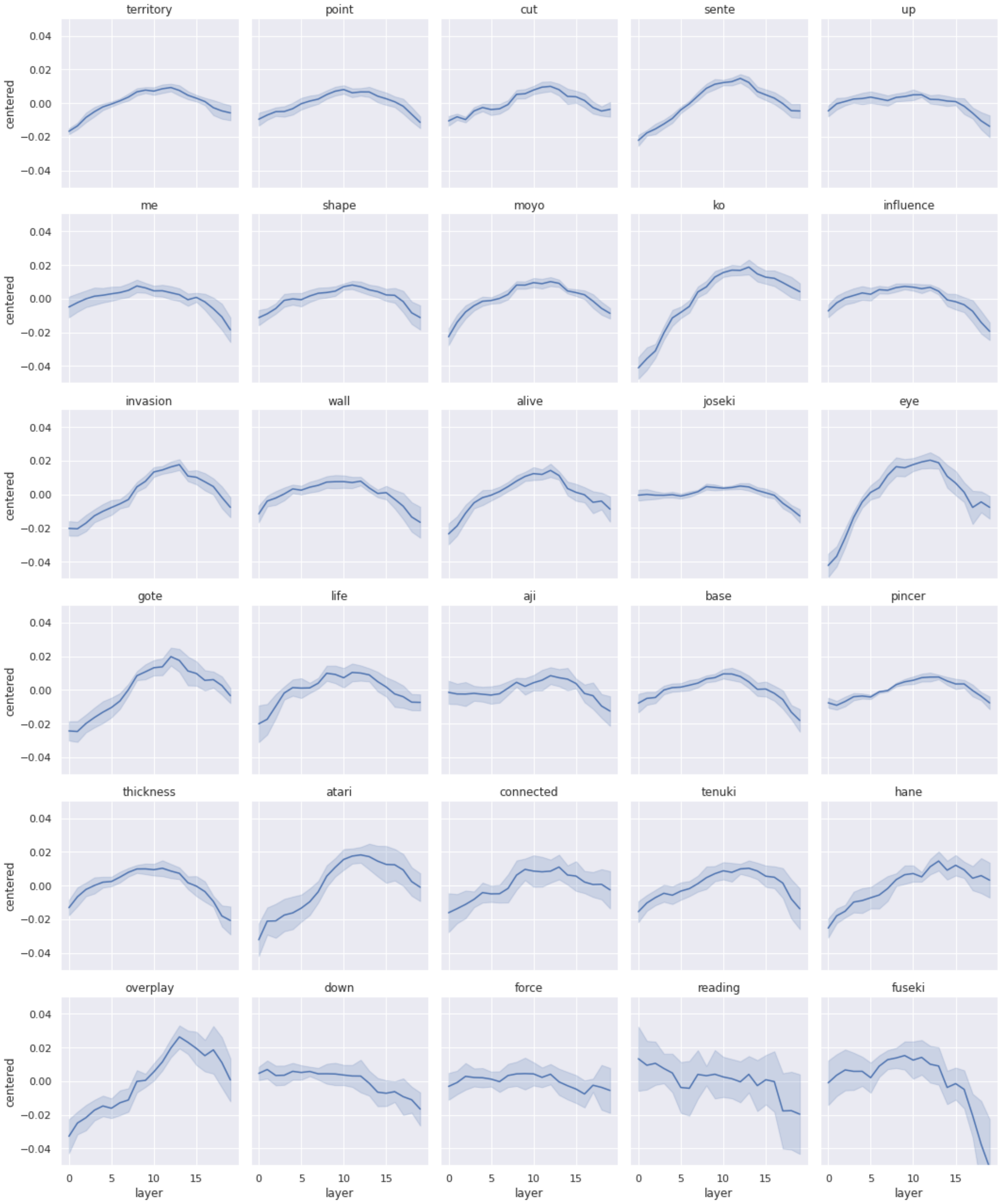


Figure 7: Plots of centered ROC AUC values for keywords in the reinforcement learning (RL) model. While noisier than the imitation learning model, probe performance still tends to peak at mid-to-late layers of the RL model and declines at the final layers. Figure 3 aggregates these results alongside pattern-based feature classifiers.

Domain Word	Imitation	Reinforcement	Rough Definition
Pincer	0.91	0.91	attack on a corner approach
Joseki	0.87	0.87	fixed local sequences of moves
Fuseki	0.85	0.84	opening
Ko	0.80	0.86	repetitive capture sequence
Base	0.76	0.77	starter eye space
Moyo	0.74	0.77	sphere of influence
Influence	0.72	0.74	long-range effect of stones
Reading	0.72	0.70	calculating an upcoming sequence
Wall	0.70	0.74	sequence of stones in a row
Thickness	0.70	0.74	strength of a group of stones
Invasion	0.70	0.72	attack on enemy territory
Atari	0.69	0.73	threat to capture
Eye	0.67	0.73	surrounded empty space
Gote	0.67	0.69	loss of initiative
Tenuki	0.66	0.68	non-local response
Hane	0.66	0.70	move that “reaches around” or bends
Overplay	0.64	0.70	overly aggressive move
Cut	0.64	0.65	block two groups from connecting
Alive	0.63	0.67	cannot be captured
Territory	0.63	0.66	controlled empty space
Aji	0.63	0.66	possibilities left in a position
Sente	0.63	0.66	initiative
Shape	0.62	0.64	quality of a group of stones
Life	0.62	0.63	inability to be captured
Connected	0.61	0.62	adjacent or nearby stones
Me	0.60	0.62	another word for *eye*
Down	0.60	0.60	toward the edge of the board
Point	0.59	0.61	specific locations on the board; or, the score
Force	0.58	0.58	requiring immediate response
Up	0.56	0.58	toward the center of the board

Table 2: ROC AUC values for all domain words in both the imitation learning and reinforcement learning models. Domain words with the highest values represent relatively straightforward corner patterns (*pincer*), while keywords with the lowest values (*force*, *up*) are polysemous with commonly used non-domain-specific meanings.

Code Synonyms Do Matter: Multiple Synonyms Matching Network for Automatic ICD Coding

Zheng Yuan[12*] **Chuanqi Tan**[2] **Songfang Huang**[2]
[1]Tsinghua University [2]Alibaba Group
yuanz17@mails.tsinghua.edu.cn
{chuanqi.tcq,songfang.hsf}@alibaba-inc.com

Abstract

Automatic ICD coding is defined as assigning disease codes to electronic medical records (EMRs). Existing methods usually apply label attention with code representations to match related text snippets. Unlike these works that model the label with the code hierarchy or description, we argue that the code synonyms can provide more comprehensive knowledge based on the observation that the code expressions in EMRs vary from their descriptions in ICD. By aligning codes to concepts in UMLS, we collect synonyms of every code. Then, we propose a multiple synonyms matching network to leverage synonyms for better code representation learning, and finally help the code classification. Experiments on the MIMIC-III dataset show that our proposed method outperforms previous state-of-the-art methods.

1 Introduction

International Classification of Diseases (ICD) is a classification and terminology that provides diagnostic codes with descriptions for diseases[1]. The task of ICD coding refers to assigning ICD codes to electronic medical records (EMRs) which is highly related to clinical tasks or systems including patient similarity learning (Suo et al., 2018), medical billing (Sonabend et al., 2020), and clinical decision support systems (Sutton et al., 2020). Traditionally, healthcare organizations have to employ specialized coders for this task, which is expensive, time-consuming, and error-prone. As a result, many methods have been proposed for automatic ICD coding since the 1990s (de Lima et al., 1998).

Recent methods treat this task as a multi-label classification problem (Xie and Xing, 2018; Li and Yu, 2020; Zhou et al., 2021), which learn deep representations of EMRs with an RNN or CNN encoder and predict codes with a multi-label classifier. Recent state-of-the-art methods propose label attention that uses the code representations as attention queries to extract the code-related representations[2] (Mullenbach et al., 2018). Following this idea, many works further propose using code hierarchical structures (Falis et al., 2019; Xie et al., 2019; Cao et al., 2020) and descriptions (Cao et al., 2020; Song et al., 2020) for better label representations.

In this work, we argue that the synonyms of codes can provide more comprehensive information. For example, the description of code *244.9* is "Unspecified hypothyroidism" in ICD. However, this code can be described in different forms in EMRs such as "low t4" and "subthyroidism". Fortunately, these different expressions can be found in the Unified Medical Language System (Bodenreider, 2004), a repository of biomedical vocabularies that contains various synonyms for all ICD codes. Therefore, we propose to leverage synonyms of codes to help the label representation learning and further benefit its matching to the EMR texts.

To model the synonym and its matching to EMR text, we further propose a **M**ultiple **S**ynonyms **M**atching **N**etwork (**MSMN**)[3]. Specifically, we first apply a shared LSTM to encode EMR texts and each synonym. Then, we propose a novel multi-synonyms attention mechanism inspired by the multi-head attention (Vaswani et al., 2017), which considers synonyms as attention queries to extract different code-related text snippets for code-wise representations. Finally, we propose using a biaffine-based similarity of code-wise text representations and code representations for classification.

We conduct experiments on the MIMIC-III dataset with two settings: full codes and top-50 codes. Results show that our method performs better than previous state-of-the-art methods.

[*] Work done at Alibaba DAMO Academy.

[1] who.int/standards/classifications/classification-of-diseases

[2] "Label" equals to "code" in some contexts of this paper.

[3] Our codes and model can be found at https://github.com/GanjinZero/ICD-MSMN.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 808 - 814
May 22-27, 2022 ©2022 Association for Computational Linguistics

2 Approach

Consider free text S (usually discharge summaries) from EMR with words $\{w_i\}_{i=1}^{N}$. The task is to assign a binary label $y_l \in \{0, 1\}$ based on S. Figure 1 shows an overview of our method.

2.1 Code Synonyms

We extend the code description l^1 by synonyms from the medical knowledge graph (i.e., UMLS Metathesaurus). We first align the code to the Concept Unique Identifiers (CUIs) from UMLS. Then we select corresponding synonyms of English terms from UMLS with the same CUIs and add additional synonyms by removing hyphens and the word "NOS" (Not Otherwise Specified). We denote the code synonyms as $\{l^2, ..., l^M\}$ in which each code synonym l^j is composed of words $\{l_i^j\}_{i=1}^{N_j}$.

2.2 Encoding

Previous works (Ji et al., 2021; Pascual et al., 2021) have shown that pretrained language models like BERT (Devlin et al., 2019) cannot help the ICD coding performance, hence we use an LSTM (Hochreiter and Schmidhuber, 1997) as our encoder. We use pre-trained word embeddings to map words w_i to $\mathbf{x}_i$. A d-layer bi-directional LSTM layer takes word embeddings as input to obtain text hidden representations $\mathbf{H} \in \mathbb{R}^h$.

$$\mathbf{H} = \mathbf{h}_1, ..., \mathbf{h}_N = \text{Enc}(\mathbf{x}_1, ..., \mathbf{x}_N) \qquad (1)$$

For code synonym l^j, we apply the same encoder with a max-pooling layer to obtain representation $\mathbf{q}^j \in \mathbb{R}^h$.

$$\mathbf{q}^j = \text{MaxPool}(\text{Enc}(\mathbf{x}_1^j, ..., \mathbf{x}_{N_j}^j)) \qquad (2)$$

2.3 Multi-synonyms Attention

To interact text with multiple synonyms, we propose a multi-synonyms attention inspired by the multi-head attention (Vaswani et al., 2017). We split $\mathbf{H} \in \mathbb{R}^{N \times h}$ into M heads $\mathbf{H}^j \in \mathbb{R}^{N \times \frac{h}{M}}$:

$$\mathbf{H} = \mathbf{H}^1, ..., \mathbf{H}^M \qquad (3)$$

Then, we use code synonyms $\mathbf{q}^j$ to query $\mathbf{H}^j$. We take the linear transformations of $\mathbf{H}^j$ and $\mathbf{q}^j$ to calculate attention scores $\alpha_l^j \in \mathbb{R}^N$. Text related to code synonym l^j can be represented by $\mathbf{H}\alpha_l^j$. We aggregate code-wise text representations $\mathbf{v}_l \in \mathbb{R}^h$ using max-pooling of $\mathbf{H}\alpha_l^j$ since the text only needs to match one of the synonyms.

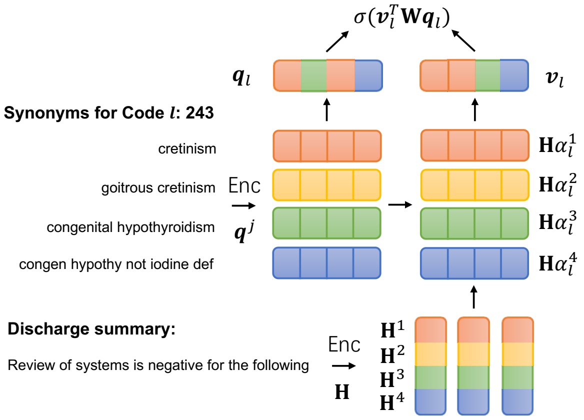


Figure 1: The architecture of our proposed MSMN. Different colors indicate different code synonyms. We also split hidden representations into different heads for multi-synonyms attention.

$$\alpha_l^j = \text{softmax}(\mathbf{W}_Q \mathbf{q}^j \cdot \tanh(\mathbf{W}_H \mathbf{H}^j)) \qquad (4)$$
$$\mathbf{v}_l = \text{MaxPool}(\mathbf{H}\alpha_l^1, ..., \mathbf{H}\alpha_l^M) \qquad (5)$$

2.4 Classification

We classify whether the text S contains code l based on the similarity between code-wise text representation $\mathbf{v}_l$ and code representation. We aggregate code synonym representations $\{\mathbf{q}^j\}$ to code representation $\mathbf{q}_l \in \mathbb{R}^h$ by max-pooling. We then propose using a biaffine transformation to measure the similarity for classification:

$$\mathbf{q}_l = \text{MaxPool}(\mathbf{q}^1, \mathbf{q}^2, ..., \mathbf{q}^M) \qquad (6)$$
$$\hat{y}_l = \sigma(\text{logit}_l) = \sigma(\mathbf{v}_l^T \mathbf{W} \mathbf{q}_l) \qquad (7)$$

Previous works (Mullenbach et al., 2018; Vu et al., 2020) classify codes via[4]:

$$\hat{y}_l = \sigma(\text{logit}_l) = \sigma(\mathbf{v}_l^T \mathbf{w}_l) \qquad (8)$$

Their work need to learn code-dependent parameters $[\mathbf{w}_l]_{l \in \mathcal{C}} \in \mathbb{R}^{\|\mathcal{C}\| \times h}$ for classification, which suffers from training rare codes. On the contrary, our biaffine function that uses $\mathbf{W}\mathbf{q}_l$ instead of $\mathbf{w}_l$ only needs to learn code-independent parameters $\mathbf{W} \in \mathbb{R}^{h \times h}$.

2.5 Training

We optimize the model using binary cross-entropy between predicted probabilities $\hat{y}_l$ and labels y_l:

$$\mathcal{L} = \sum_{l \in \mathcal{C}} -y_l \log(\hat{y}_l) - (1 - y_l) \log(1 - \hat{y}_l) \qquad (9)$$

[4] We omit the biases in all equations for simplification.

	Train	Dev	Test
MIMIC-III Full			
# Doc.	47,723	1,631	3,372
Avg # words per Doc.	1,434	1,724	1,731
Avg # codes per Doc.	15.7	18.0	17.4
Total # codes	8,692	3,012	4,085
MIMIC-III 50			
# Doc.	8,066	1,573	1,729
Avg # words per Doc.	1,478	1,739	1,763
Avg # codes per Doc.	5.7	5.9	6.0
Total # codes	50	50	50

Table 1: Statistics of MIMIC-III dataset under full codes and top-50 codes settings.

Parameters	Full	Top 50
Emb. dim.	100	100
Emb. dropout	0.2	0.2
LSTM Layer (d)	2	1
LSTM hidden dim.	256	512
LSTM output dim. (h)	512	512
Synonyms count (M)	4	8
Rep. dropout	0.2	0.2
R-Drop weight	5.0	5.0
Epoch	20	20
Peak lr.	5e-4	5e-4
Batch size	16	16
Adam ϵ	1e-8	1e-8
Weight decay	0.01	0.01
Clipping grad.	1.0	1.0

Table 2: Hyper-parameters used for training MIMIC-III full setting and MIMIC-III 50 setting.

3 Experiments

3.1 Dataset

MIMIC-III dataset (Johnson et al., 2016) contains deidentified discharge summaries with human-labeled ICD-9 codes. We list the document counts, average word counts per document, average codes counts per document, and total codes of the MIMIC-III dataset in Table 1. We use the same splits with previous works (Mullenbach et al., 2018; Vu et al., 2020) with two settings as full codes (MIMIC-III full) and top-50 frequent codes (MIMIC-III 50). We follow the preprocessing of Xie et al. (2019) and Vu et al. (2020) to truncate discharge summaries at 4,000 words. We measure the results using macro AUC, micro AUC, macro F_1, micro F_1 and precision@k ($k = 5$ for MIMIC-III 50, 8 and 15 for MIMIC-III full).

3.2 Implementation Details

We sample $M = 4$ and 8 synonyms per code for MIMIC-III full and MIMIC-III 50 respectively. We sample synonyms fully randomly from the synonyms set. If some ICD codes do not have enough synonyms, we just repeat these synonyms. We use the same word embeddings as Vu et al. (2020) which are pretrained on the MIMIC-III discharge summaries using CBOW (Mikolov et al., 2013) with a hidden size of 100. We apply R-Drop with $\alpha = 5$ (Liang et al., 2021) to regularize the model to prevent over-fitting. We apply the dropout with a ratio of 0.2 after the word embedding layer and before the classification layer. For text encoding, we add a linear layer upon the LSTM layer (the output dimension of the linear layer refers to LSTM output dim. in Table 2). We train MSMN with AdamW (Loshchilov and Hutter, 2019) with a linear learning rate decay. We optimize the threshold of classification using the development set. For the MIMIC-III 50 setting, we train with one 16GB NVIDIA-V100 GPU. For the MIMIC-III full setting, we train with 8 32GB NVIDIA-V100 GPUs. We list the detailed training hyper-parameters in Table 2.

3.3 Baselines

CAML (Mullenbach et al., 2018) uses CNN to encode texts and proposes label attention for coding.
MSATT-KG (Xie et al., 2019) applies multi-scale attention and GCN to capture codes relations.
MultiResCNN (Li and Yu, 2020) encodes text using multi-filter residual CNN.
HyperCore (Cao et al., 2020) embeds ICD codes into the hyperbolic space to utilize code hierarchy and uses GCN to leverage the code co-occurrence.
LAAT & **JointLAAT** (Vu et al., 2020) propose a hierarchical joint learning mechanism to relieve the imbalanced labels, which is our main baseline since it is most similar to our work.

3.4 Main Results

Table 3 and 4 show the main results under the MIMIC-III full and MIMIC-III 50 settings, respectively. Under the full setting, our MSMN achieves 95.0 (+2.0), 99.2 (+0.0), 10.3 (-0.4), 58.4 (+0.9), 75.2 (+1.4), and 59.9 (+0.8) in terms of macro-AUC, micro-AUC, macro-F_1, micro-F_1, P@8, and P@15 respectively (parentheses shows the differences against previous best results), which shows that MSMN obtains state-of-the-art results in most metrics. Under the top-50 codes setting, MSMN performs better than LAAT in all metrics and achieves state-of-the-art scores of 92.8 (+0.3), 94.7 (+0.1), 68.3 (+1.7), 72.5 (+0.9), 68.0 (+0.5)

	AUC		F_1		Precision@N	
	Macro	Micro	Macro	Micro	P@8	P@15
CAML (Mullenbach et al., 2018)	89.5	98.6	8.8	53.9	70.9	56.1
MSATT-KG (Xie et al., 2019)	91.0	**99.2**	9.0	55.3	72.8	58.1
MultiResCNN (Li and Yu, 2020)	91.0	98.6	8.5	55.2	73.4	58.4
HyperCore (Cao et al., 2020)	93.0	98.9	9.0	55.1	72.2	57.9
LAAT (Vu et al., 2020)	91.9	98.8	9.9	57.5	73.8	59.1
JointLAAT (Vu et al., 2020)	92.1	98.8	**10.7**	57.5	73.5	59.0
MSMN	**95.0**	**99.2**	10.3	**58.4**	**75.2**	**59.9**

Table 3: Results on the MIMIC-III full test set.

	AUC		F_1		
	Macro	Micro	Macro	Micro	P@5
CAML	87.5	90.9	53.2	61.4	60.9
MSATT-KG	91.4	93.6	63.8	68.4	64.4
MultiResCNN	89.9	92.8	60.6	67.0	64.1
HyperCore	89.5	92.9	60.9	66.3	63.2
LAAT	92.5	94.6	66.6	71.5	67.5
JointLAAT	92.5	94.6	66.1	71.6	67.1
MSMN	**92.8**	**94.7**	**68.3**	**72.5**	**68.0**

Table 4: Results on the MIMIC-III 50 test set.

	AUC		F_1		
	Macro	Micro	Macro	Micro	P@5
$M = 1$	92.1	94.2	67.4	71.0	67.0
$M = 2$	92.6	94.6	67.6	71.7	67.2
$M = 4$	**92.8**	**94.7**	67.9	71.9	67.7
$\underline{M = 8}$	**92.8**	**94.7**	**68.3**	**72.5**	**68.0**
$M = 16$	92.5	94.6	66.9	71.5	67.6
$\underline{\mathbf{v}_l^T \mathbf{W} \mathbf{q}_l}$	**92.8**	**94.7**	**68.3**	**72.5**	**68.0**
$\mathbf{v}_l^T \mathbf{q}_l$	92.5	94.5	67.1	71.2	67.1
$\mathbf{v}_l^T \mathbf{w}_l$	91.5	94.1	65.1	70.8	66.3

Table 5: Results of different settings including synonyms counts and scoring functions on MIMIC-III 50 dataset. Underlined setting denotes the default parameters used in MSMN.

on macro-AUC, micro-AUC, macro-F_1, micro-F_1, and P@5, respectively. We notice that the macro F_1 has a large variance in every epoch under the MIMIC-III full setting since it is more sensitive in a long tail problem.

3.5 Discussion

To explore the influence of leveraging different numbers of code synonyms, we search M among $\{1, 2, 4, 8, 16\}$ on the MIMIC-III 50 dataset. Results are shown in Table 5. Compared with $M = 1$ that we only use the original ICD code descriptions, leveraging more synonyms from UMLS consistently improves the performance. Using $M = 4, 8$ achieves the best performance in terms of AUC, and $M = 8$ achieves the best performance in terms of F_1 and P@5. In addition, the median and mean count of UMLS synonyms are 5.0 and 5.4 respectively, which echoes why the results of $M = 4$ or 8 are better.

To evaluate the effectiveness of our proposed biaffine-based similarity function, we compare it with the baseline LAAT in Table 5. We also provide a simple function by removing $\mathbf{W}$ to $\mathbf{v}_l^T \mathbf{q}_l$ in Equation 7. Results show that the biaffine-based similarity scoring performs best among others.

To better understand what MSMN learns from the multi-synonyms attention, we plot the synonym representations $\mathbf{q}^j$ under MIMIC-III 50 setting via t-SNE (van der Maaten and Hinton, 2008) in Figure 2. We observe for some codes like *585.9* ("chronic kidney diseases"), all synonym representations cluster together, which indicates that synonyms extract similar text snippets. However, codes like *410.71* ("subendocardial infarction initial episode of care" or "subendo infarct, initial") and *403.90* ("hypertensive chronic kidney disease, unspecified, with chronic kidney disease stage i through stage iv" or "unspecified orhy kid w cr kid i iv") with very different synonyms learn different representations, which benefits to match different text snippets. Furthermore, we observe it has similar representations for sibling codes *37.22* ("left heart cardiac catheterization") and *37.23* ("rt/left heart card cath"), which indicates the model can also implicitly capture the code hierarchy.

3.6 Memory Complexity

The memory usage of our proposed MSMN is dominated by Equation 4 and Equation 5. We suppose batch size as B, word count as N, label count as C and synonyms count as M. Calculating Equation 4 for all j simultaneously requires calculating Einstein summation (Daniel et al., 2018) among tensors with shape $B \times N \times h$ and shape

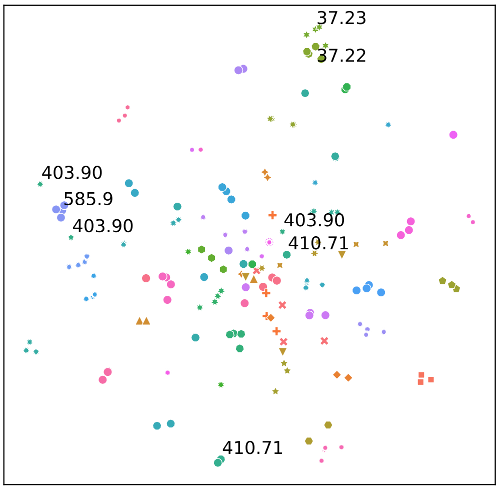


Figure 2: T-SNE visualization of code synonym representations learned from MIMIC-III 50.

$C \times M \times h$ to shape $B \times C \times N \times M$. Calculating Equation 5 requires calculating Einstein summation among tensors with shape $B \times N \times h$ and shape $B \times C \times N \times M$ to shape $B \times C \times h \times M$. The memory complexities of these two equations are linearly proportional to M.

4 Related Work

Automatic ICD coding is an important task in the medical NLP community. Earlier works use machine learning methods for coding (Larkey and Croft, 1996; Pestian et al., 2007; Perotte et al., 2014). With the development of neural networks, many recent works consider ICD coding as a multi-label text classification task. They usually apply RNN or CNN to encode texts and use the label attention mechanism to extract and match the most relevant parts for classification. The label attention relies on the label representations as attention queries. Li and Yu (2020); Vu et al. (2020) randomly initialize the label representations which ignore the code semantic information. Cao et al. (2020) use the average of word embeddings as label representations to leverage the code semantic information. Xie et al. (2019); Cao et al. (2020) use GCN to fuse hierarchical structures of ICD codes for label representations. Compared with previous works, we use synonyms instead of a single description to represent the code, which can provide more comprehensive expressions of codes.

Biomedical entity linking is a related task to automatic ICD coding. The task requires standardizing given terms to a pre-defined concept dictionary. There are two differences between biomedical entity linking and automatic ICD coding: (1) They have different target concepts. ICD coding map EMRs to ICD codes, while biomedical entity linking usually map terms to a larger dictionary like SNOMED-CT or UMLS. (2) They have different input formats. Entity linking task has labeled entities in texts, while ICD coding only provides texts. Synonyms have also been used in biomedical entity linking (Sung et al., 2020; Yuan et al., 2022). BioSYN (Sung et al., 2020) uses marginalization to sum the probabilities of all synonyms as the similarity between a term and a concept. However, we consider multi-synonyms attention to extracting different parts of clinical texts to interact with synonyms.

5 Conclusions

In this paper, we propose MSMN to leverage code synonyms from UMLS to improve the automatic ICD coding. Multi-synonyms attention is proposed for extracting different related text snippets for code-wise text representations. We also propose a biaffine transformation to calculate similarities among texts and codes for classification. Experiments show that MSMN outperforms previous methods with label attention and achieves state-of-the-art results in the MIMIC-III dataset. Ablation studies show the effectiveness of multi-synonyms attention and biaffine-based similarity.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments and suggestions. We thank Fuli Luo, Shengxuan Luo, Hongyi Yuan, Xu Chen, and Jiayu Li for their help. This work was supported by Alibaba Group through Alibaba Research Intern Program.

References

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Shengping Liu, and Weifeng Chong. 2020. Hypercore: Hyperbolic and co-graph representation for automatic icd coding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3105–3114.

G Daniel, Johnnie Gray, et al. 2018. Opt_einsum-a python package for optimizing contraction order for einsum-like expressions. *Journal of Open Source Software*, 3(26):753.

Luciano RS de Lima, Alberto HF Laender, and Berthier A Ribeiro-Neto. 1998. A hierarchical approach to the automatic categorization of medical documents. In *Proceedings of the seventh international conference on Information and knowledge management*, pages 132–139.

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

Matus Falis, Maciej Pajak, Aneta Lisowska, Patrick Schrempf, Lucas Deckers, Shadia Mikhael, Sotirios Tsaftaris, and Alison O'Neil. 2019. Ontological attention ensembles for capturing semantic concepts in icd code prediction from clinical text. In *Proceedings of the Tenth International Workshop on Health Text Mining and Information Analysis (LOUHI 2019)*, pages 168–177.

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

Shaoxiong Ji, Matti Hölttä, and Pekka Marttinen. 2021. Does the magic of bert apply to medical code assignment? a quantitative study. *Computers in Biology and Medicine*, 139:104998.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Leah S Larkey and W Bruce Croft. 1996. Combining classifiers in text categorization. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 289–297.

Fei Li and Hong Yu. 2020. Icd coding from clinical text using multi-filter residual convolutional neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8180–8187.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. In *NeurIPS*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1101–1111, New Orleans, Louisiana. Association for Computational Linguistics.

Damian Pascual, Sandro Luck, and Roger Wattenhofer. 2021. Towards BERT-based automatic ICD coding: Limitations and opportunities. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 54–63, Online. Association for Computational Linguistics.

Adler Perotte, Rimma Pivovarov, Karthik Natarajan, Nicole Weiskopf, Frank Wood, and Noémie Elhadad. 2014. Diagnosis code assignment: models and evaluation metrics. *Journal of the American Medical Informatics Association*, 21(2):231–237.

John P. Pestian, Chris Brew, Pawel Matykiewicz, DJ Hovermale, Neil Johnson, K. Bretonnel Cohen, and Wlodzislaw Duch. 2007. A shared task involving multi-label classification of clinical free text. In *Biological, translational, and clinical language processing*, pages 97–104, Prague, Czech Republic. Association for Computational Linguistics.

Aaron Sonabend, Winston Cai, Yuri Ahuja, Ashwin Ananthakrishnan, Zongqi Xia, Sheng Yu, and Chuan Hong. 2020. Automated icd coding via unsupervised knowledge integration (unite). *International journal of medical informatics*, 139:104135.

Congzheng Song, Shanghang Zhang, Najmeh Sadoughi, Pengtao Xie, and Eric Xing. 2020. Generalized zero-shot text classification for icd coding. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4018–4024. International Joint Conferences on Artificial Intelligence Organization. Main track.

Mujeen Sung, Hwisang Jeon, Jinhyuk Lee, and Jaewoo Kang. 2020. Biomedical entity representations with synonym marginalization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3641–3650, Online. Association for Computational Linguistics.

Qiuling Suo, Fenglong Ma, Ye Yuan, Mengdi Huai, Weida Zhong, Jing Gao, and Aidong Zhang. 2018. Deep patient similarity learning for personalized healthcare. *IEEE transactions on nanobioscience*, 17(3):219–227.

Reed T Sutton, David Pincock, Daniel C Baumgart, Daniel C Sadowski, Richard N Fedorak, and Karen I Kroeker. 2020. An overview of clinical decision support systems: benefits, risks, and strategies for success. *NPJ digital medicine*, 3(1):1–10.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

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

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. 2020. A label attention model for icd coding from clinical text. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3335–3341. Main track.

Pengtao Xie and Eric Xing. 2018. A neural architecture for automated icd coding. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1066–1076.

Xiancheng Xie, Yun Xiong, Philip S Yu, and Yangyong Zhu. 2019. Ehr coding with multi-scale feature attention and structured knowledge graph propagation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 649–658.

Zheng Yuan, Zhengyun Zhao, Haixia Sun, Jiao Li, Fei Wang, and Sheng Yu. 2022. Coder: Knowledge-infused cross-lingual medical term embedding for term normalization. *Journal of Biomedical Informatics*, page 103983.

Tong Zhou, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Kun Niu, Weifeng Chong, and Shengping Liu. 2021. Automatic icd coding via interactive shared representation networks with self-distillation mechanism. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5948–5957.

CoDA21: Evaluating Language Understanding Capabilities of NLP Models With Context-Definition Alignment

Lütfi Kerem Senel, Timo Schick and **Hinrich Schütze**
Center for Information and Language Processing (CIS), LMU Munich, Germany
lksenel@gmail.com, schickt@cis.lmu.de

Abstract

Pretrained language models (PLMs) have achieved superhuman performance on many benchmarks, creating a need for harder tasks. We introduce CoDA21 (Context Definition Alignment), a challenging benchmark that measures natural language understanding (NLU) capabilities of PLMs: Given a definition and a context each for k words, but not the words themselves, the task is to align the k definitions with the k contexts. CoDA21 requires a deep understanding of contexts and definitions, including complex inference and world knowledge. We find that there is a large gap between human and PLM performance, suggesting that CoDA21 measures an aspect of NLU that is not sufficiently covered in existing benchmarks.[1]

1 Introduction

Increasing computational power along with the design and development of large and sophisticated models that can take advantage of enormous corpora has drastically advanced NLP. For many tasks, finetuning pretrained transformer-based language models (Vaswani et al., 2017; Devlin et al., 2019; Radford et al., 2018) has improved the state of the art considerably. Language models acquire knowledge during pretraining that is utilized during task-specific finetuning. On benchmarks that were introduced to encourage development of models that do well on a diverse set of NLU tasks (e.g., GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019)), these models now achieve superhuman performance (He et al., 2020). The pretrain-then-finetune approach usually requires a great amount of labeled data, which is often not available or expensive to obtain, and results in specialized models that can perform well only on a single task. Recently, it was shown that generative language models can be applied to many tasks without finetuning when the task is formulated as text generation and the PLM is queried with a natural language prompt (Radford et al., 2019; Brown et al., 2020).

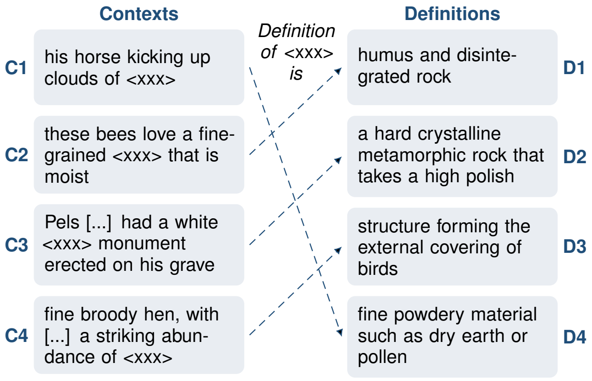

Contexts
Definitions
Definition of <xxx> is
C1
his horse kicking up clouds of <xxx>
C2
these bees love a fine-grained <xxx> that is moist
C3
Pels [...] had a white <xxx> monument erected on his grave
C4
fine broody hen, with [...] a striking abundance of <xxx>
humus and disintegrated rock
D1
a hard crystalline metamorphic rock that takes a high polish
D2
structure forming the external covering of birds
D3
fine powdery material such as dry earth or pollen
D4

Figure 1: The CoDA21 task is to find the correct alignment between contexts and definitions: **C1-D4**, **C2-D1**, **C3-D2**, **C4-D3**. The target words (**C1**:"dust", **C2**:"soil", **C3**:"marble", **C4**:"feathers"; not provided to the model) are replaced with a placeholder <xxx>.

Motivated by recent progress in zero-shot learning with generative models as well as the need for more challenging benchmarks that test language understanding of language models, we introduce CoDA21 (**Co**ntext **D**efinition **A**lignment), a difficult benchmark that measures NLU capabilities of PLMs for the English language. Given a definition and a context each for k words, but not the words themselves, the task is to align the k definitions with the k contexts. In other words, for each definition, the context in which the defined word is most likely to occur has to be identified. This requires (i) understanding the definitions, (ii) understanding the contexts, and (iii) the ability to match the two. Since the target words are not given, a model must be able to distinguish subtle meaning differences between different contexts/definitions to be successful. To illustrate the difficulty of the task, Figure 1 shows a partial example for $k = 4$ (see Table 5 in the supplementary for the full ex-

[1] Our dataset and code are available at https://github.com/lksenel/CoDA21

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 815 - 824
May 22-27, 2022 ©2022 Association for Computational Linguistics

ample). We see that both complex inference (e.g., <XXX> can give rise to a cloud by being kicked up ⇒ <XXX> must be dry ⇒ <XXX> can be dust, but not soil) and world knowledge (what materials are typical for monuments?) are required for CoDA21.

We formulate the alignment task as a text prediction task and evaluate, without finetuning, three PLMs on CoDA21: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and GPT-2 (Radford et al., 2019). Poor performance of the PLMs and a large gap between human and PLM performance suggest that CoDA21 is an important benchmark for designing models with better NLU capabilities.

2 CoDA21

2.1 Dataset

We construct CoDA21 by first deriving a set $\mathcal{G}$ of *synset groups* $\{G_1, G_2, \ldots\}$ from Wordnet (Miller, 1995). A synset group G_i is a group of synsets whose meanings are close enough to be difficult to distinguish (making the task hard), but not so close that they become indistinguishable for human and machine. In a second step, each synset group G_i is converted into a *CoDA21 group* G_i^+ – a set of triples, each consisting of the synset, its definition, and a corpus context. A CoDA21 group can be directly used for one instance of the CoDA21 task.

Synset groups. Each synset group G consists of $5 \leq k \leq 10$ synsets. To create a synset group, we start with a *parent synset* $\hat{s}$ and construct a co-hyponym group $\bar{G}(\hat{s})$ of its children:

$$\bar{G}(\hat{s}) = \{s \mid s < \hat{s}, s \notin D\}$$

where $<$ is the hyponymy relation between synsets and D is the set of synsets that have already been added to a synset group. The intuition for grouping synsets with a common parent is that words sharing a hypernym are difficult to distinguish (as opposed to randomly selected words).

We iterate $\hat{s}$ through all nouns and verbs in WordNet. At each iteration, we get all hyponyms of $\hat{s}$ that have not been previously added to a synset group; not reusing a synset ensures that different CoDA21 subtasks are not related and so no such relationships can be exploited.

We extract synset groups from co-hyponym groups by splitting them into multiple chunks of size k. In an initial exploration, we found that the task is hard to solve for human subjects if two closely related hyponyms are included, e.g., "clementine" and "tangerine". We therefore employ clustering to assemble a set of mutually dissimilar hyponyms. We first compute a sentence embedding for each hyponym definition using the *stsb-distilbert-base* Sentence Transformer model[2]. We then cluster the embeddings using complete-link clustering, combining the two most dissimilar clusters in each step. We stop merging before the biggest cluster exceeds the maximum group size ($k = 10$) or before the similarity between the last two combined clusters exceeds the maximum similarity ($\theta = 0.8$). The largest cluster G is added to the set $\mathcal{G}$ of synset groups. We then iterate the steps of (i) removing the synsets in the previous largest cluster G from $\bar{G}(\hat{s})$ and (ii) running complete-link clustering and adding the resulting largest cluster G to $\mathcal{G}$ until fewer than five synsets remain in $\bar{G}(\hat{s})$ or no cluster can be formed whose members have a similarity of less than θ.

CoDA21 groups. For each synset s, we extract its definition $d(s)$ from WordNet and a context $c(s)$ in which it occurs from SemCor[3] (Miller et al., 1994). SemCor is an English corpus tagged with WordNet senses. Let $C(s)$ be the set of contexts of s in SemCor. If $|C(s)| > 1$, we use as $c(s)$ the context in which *bert-base-uncased* predicts **s** with the highest log probability when it is masked, where **s** is the word tagged with the sense s[4] – this favors contexts that are specific to the meaning of the synset. Finally, we convert each synset group G_i in $\mathcal{G}$ to a CoDA21 group G_i^+:

$$G_i^+ = \{(s_j, d(s_j), c(s_j)) \mid s_j \in G_i\}$$

That is, a CoDA21 group G_i^+ is a set of triples of sense, definition and context. In PLM evaluation, each CoDA21 group G_i^+ gives rise to one context-definition alignment subtask.

We name the resulting dataset *CoDA21-noisy-hard*: *noisy* because if $|C(s)|$ is small, the selected context may not be informative enough to identify the matching definition; *hard* because the synsets in a CoDA21 group are taxonomic sisters, generally with similar meanings despite the clustering-based limit on definition similarity. We construct a *clean* version of the dataset by only using synsets with $|C(s)| \geq 5$. We also construct an *easy* version by

[2] https://huggingface.co/sentence-transformers/stsb-distilbert-base

[3] We do not consider synsets without contexts in SemCor.

[4] We average the probabilities when **s** is tokenized to multiple tokens.

Dataset	noun		verb	
	# of G	USC	# of G	USC
CoDA21-*clean-hard*	106	740	102	711
CoDA21-*clean-easy*	274	1999	103	758
CoDA21-*noisy-hard*	691	4633	350	2527
CoDA21-*noisy-easy*	1188	8910	370	2766

Table 1: CoDA21 group (G) statistics, USC: Unique Synset Count

taking the "hyponym grandchildren" s of a parent synset $\hat{s}$ ($s < l \wedge l < \hat{s}$) instead of its hyponym children. This reduces the similarity of synsets in a CoDA21 group, making the task easier. Table 1 gives dataset statistics.

2.2 Alignment

Recall the CoDA21 task: given a definition and a context each for k words (but not the words themselves), align the k definitions with the k contexts. That is, we are looking for a bijective function (a one-to-one correspondence) between definitions and contexts. Our motivation in designing the task is that we want a hard task (which can guide us in developing stronger natural language understanding models), but also a task that is solvable by humans. Our experience is that humans can at least partially solve the task by finding a few initial "easy" context-definition matches, removing them from the definition/context sets and then match the smaller remaining number of definitions/contexts.

The number of context-definition pairs scales quadratically ($O(k^2)$) with k and the number of alignments factorially ($O(k!)$). We restrict k to $k \leq 10$ to make sure that we do not run into computational problems and that humans do not find the task too difficult.

In order to connect contexts to definitions without using the target words, we replace the target words by a made-up word. This setup resembles the incidental vocabulary acquisition process in humans. Let t be a target word, c a context in which t occurs and m a made-up word. To test PLMs on CoDA21, we use the following pattern[5]:

$$Q(c, m) = c_m \text{ Definition of } m \text{ is}$$

where c_m is c with occurrences of t replaced by m.

We calculate the *match score* of a context-definition pair (c, d) as $\log P(d \mid Q(c, m))$, i.e., log generation probability of the definition d conditioned on $Q(c, m)$. Our objective is to maximize the sum of the k match scores in an alignment. We find the best alignment by exhaustive search. Accuracy for a CoDA21 group G_i^+ is then the accuracy of its best alignment, i.e., the number of contexts in G_i^+ that are aligned with the correct definition, divided by the total number of contexts $|G_i^+|$.

2.3 Baselines

We calculate $P(d \mid Q(c, m))$ for a masked language model (MLM) M and an autoregressive language model (ALM) A as follows:

$$P_M(d \mid Q') = \textstyle\prod_{i=1}^{|d|} P(d_i \mid Q', d_{-i})$$
$$P_A(d \mid Q') = \textstyle\prod_{i=1}^{|d|} P(d_i \mid Q', d_1, \ldots, d_{i-1})$$

where $Q' = Q(c, m)$, d_i is the i^{th} word in definition d and d_{-i} is the definition with the i^{th} word masked.

We evaluate the MLMs BERT and RoBERTa and the ALM GPT-2. We experiment with both base and large versions of BERT and RoBERTa and with all four sizes of GPT-2 (small, medium, large, xl), for a total of eight models, to investigate the effect of model size on performance.

The made-up word m should ideally be unknown so that it does not bias the PLM in any way. However, there are no truly unknown words for the models we investigate due to the word-piece tokenization they apply to the input. Any made-up word that is completely meaningless to humans will have a representation in the models' input space based on its tokenization. To minimize the risk that the meaning of the made-up word may bias the model, we use m = *bkatuhla*, a word with an empty search result on Google that most likely never appeared in the models' pretraining corpora.

In addition to PLMs, we also evaluate 2 recent sentence transformer models[6] (Reimers and Gurevych, 2019), *paraphrase-mpnet-base-v2* (mpnet) and *paraphrase-MiniLM-L6-v2* (MiniLM), and fastText static embeddings[7](Mikolov et al., 2018). To calculate the match score of a context-definition pair, we first remove the target word from the context and represent contexts and definitions as vectors. For sentence transformers, we obtain these vectors by simply encoding the input sentences. For fastText, we average the vectors of the

[5] When the target word is a verb (i.e., verb subset of a CoDA21 dataset), we add "to" at the end of our pattern.

[6] https://www.sbert.net/docs/pretrained_models.html

[7] We use the *crawl-300d-2M-subword* model from https://fasttext.cc/docs/en/english-vectors.html

words in contexts and definitions. We then calculate the match score as the cosine similarity of context and definition vectors.

3 Results

Table 2 presents average accuracy of the investigated models on the four CoDA21 datasets. As can be seen, fastText performs only slightly better than random. MLMs also perform better than random chance by only a small margin. This poor performance can be partly explained by the generation style setup we use, which is not well suited for masked language models. Even the smallest GPT-2 model performs considerably better than RoBERTa-large, the best performing MLM. Performance generally improves with model size. GPT-2$_{xl}$ achieves the best results among the LMs on almost all datasets. Interestingly, sentence transformer *all-mpnet-base-v2* performs comparably to GPT-2$_{xl}$ on most datasets despite its simple, similarity based matching compared to generation based matching of GPT-2 models. *Based on this observation it can be argued that current state-of-the-art language models fail to perform complex, multi-step reasoning and inference which are necessary to solve the CoDA21 tasks.* Overall, MLMs perform slightly better on verbs than nouns while the converse is true for GPT-2. As expected, all models perform better on the *easy* datasets. Performance on *noisy* and *clean* datasets are comparable; this indicates that our contexts are of high quality even for the synsets with only a few contexts.

Human performance on CoDA21. We asked two NLP PhD students[8] to solve the task on S20, a random sample of size 20 from the noun part of CoDA21-*clean-easy*. Table 2 shows results on S20 for these two subjects and our models. Human performance is 0.86 – compared to 0.48 for GPT-2$_{xl}$, the best performing model. This difference indicates that there is a large gap in NLU competence between current language models and humans and that CoDA21 is a good benchmark to track progress on closing that gap.

To investigate the **effect of the made-up word** m, we experiment with several other words on the noun part of CoDA21-*clean-easy*. Specifically, we investigate another nonce word "opyatzel", a single letter "x" and two frequent words "orange" and "cloud". Table 3 shows the results of the models for different made-up words. MLMs do not show significant variability in performance, and perform comparably poor for all words tried. GPT2 versions, which perform considerably better than MLMs on CoDA21, perform similarly for the two nonce words and single letter "x", which do not have a strong meaning. Their performance drops significantly when the two frequent words are used as the made-up word, due to the effect of prior knowledge models have about these words.

	clean hard		clean easy		noisy hard		noisy easy		S20
Model	N	V	N	V	N	V	N	V	N
BERT$_b$	.20	.21	.22	.25	.21	.22	.22	.24	.24
BERT$_l$	.22	.22	.19	.21	.19	.20	.20	.20	.22
RoBERTa$_b$	.24	.26	.26	.32	.25	.25	.28	.27	.29
RoBERTa$_l$	.26	.30	.30	.30	.27	.29	.30	.33	.29
GPT-2$_s$	.31	.32	.42	.40	.35	.32	.40	.36	.35
GPT-2$_m$	.37	.35	.45	.39	.38	.35	.43	.39	.39
GPT-2$_l$	.38	.34	.47	**.42**	.39	**.37**	**.46**	.41	.47
GPT-2$_{xl}$	**.42**	.36	**.49**	**.42**	**.40**	.36	**.46**	**.43**	.48
mpnet	**.42**	**.39**	.48	**.42**	**.40**	**.37**	**.46**	.40	.51
MiniLM	.35	.34	.40	.36	.34	.30	.38	.32	.34
fastText	.18	.17	.20	.20	.18	.18	.18	.18	.17
Random	.15	.15	.14	.14	.16	.15	.14	.14	.14
Human	–	–	–	–	–	–	–	–	**.86**

Table 2: Average accuracy on the noun (N) and verb (V) subsets of CoDA21 for eight PLMs, two sentence transformers, fastText embeddings and (on S20) for humans

Model	bkatuhla	opyatzel	x	cloud	orange
BERT$_b$	.22	.22	.22	.23	.22
BERT$_l$	.19	.19	.20	.20	.19
RoBERTa$_b$	.26	.27	.26	.28	.28
RoBERTa$_l$	.30	.30	.29	.30	.29
GPT-2$_s$	.42	.43	.41	.39	.39
GPT-2$_m$	.45	.42	.43	.40	.41
GPT-2$_l$	.47	.46	.47	.41	.42
GPT-2$_{xl}$	.49	.44	.45	.40	.41

Table 3: Average accuracy of eight PLMs on the noun subsets of CoDA21-*clean-easy* using various words as the made-up word.

To investigate the **effect of the pattern**, we compared our pattern $Q(c, m)$ with two alternative patterns by evaluating GPT-2$_{xl}$ on the noun part of CoDA21-*clean-easy*. Patterns and the evaluation results are shown in Table 4. The results suggest that the effect of the pattern on performance is minimal.

Effect of the alignment setup. We constructed CoDA21 as an alignment dataset which uses the fact that matching between the definitions and contexts is one-to-one. This setup makes the task

[8] Both are proficient (though not native) English speakers.

Pattern	Acc
c_m Definition of m is	0.49
c_m m is defined as	0.51
c_m m is	0.49

Table 4: Effect of the pattern on the performance of GPT2-$_{xl}$ on the noun part of CoDA21-*clean-easy*

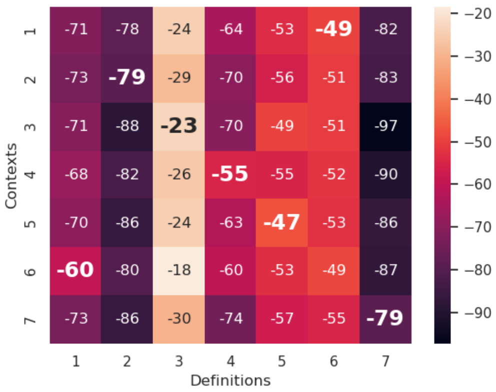


Figure 2: Match scores from GPT2-xl model for the context definition pairs for the sample given in Table 5. Match scores shown in bold correspond the context-definition pairs that are in the predicted alignment by the model that yields maximum total match score.

more intuitive and manageable for humans. However, context-definition match scores can be used to evaluate models on CoDA21 samples also without the alignment setup by simply picking context-definition pairs with the highest match score for each definition. We additionally evaluated GPT-2$_{xl}$ model on CoDA21-*clean-easy* dataset using this simple matching approach which yielded 0.38 average accuracy compared to the 0.49 accuracy achieved with the alignment setup. This result suggests that language models can also make use of the alignment style evaluation, similar to humans.

Table 5 (in the Appendix) presents a sample of size 7 from the noun part of the CoDA21-*clean-easy* dataset. Figure 2 displays all 49 match scores of the context-definition pairs for this sample obtained using GPT-2$_{xl}$. 5 of the 7 definitions (2,3,4,5,7) are matched with correct contexts with the alignment setup while 4 definitions (4,5,6,7) are matched correctly for the simple matching setup. Alignment setup enabled the model to match second and third definitions with their corresponding contexts even though their match scores are not the highest ones.

To get a better sense of why the task is hard for PLMs, we give an example, from the CoDA21 subtask in Figure 1 (also Figure 2 and Table 5 refer to the same subtask), of a context-definition match that is scored highly by GPT-2$_{xl}$, but is not correct. **Context:** "these bees love a fine-grained <XXX> that is moist". **Definition:** "fine powdery material such as dry earth or pollen". (context 6 and definition 1 in Figure 2) GPT-2$_{xl}$ most likely gives a high score because it has learned that *bees* and *pollen* are associated. It does not understand that the mutual exclusivity of "moist" and "powdery" makes this a bad match.

4 Related Work

There are many datasets (Levesque et al., 2012; Rajpurkar et al., 2016; Williams et al., 2018) for evaluating language understanding of models. Many adopt a text prediction setup: Lambada (Paperno et al., 2016) evaluates the understanding of discourse context, StoryCloze (Mostafazadeh et al., 2016) evaluates commonsense knowledge and so does HellaSwag (Zellers et al., 2019), but examples were adversarially mined. LAMA (Petroni et al., 2019) tests the factual knowledge contained in PLMs. In contrast to this prior work, CoDA21 goes beyond prediction by requiring the matching of pieces of text. WIC (Pilehvar and Camacho-Collados, 2019) is also based on matching, but CoDA21 is more complex (multiple contexts/definitions as opposed to a single binary match decision) and is not restricted to ambiguous words. WNLaMPro (Schick and Schütze, 2020) evaluates knowledge of subordinate relationships between words, and WDLaMPro (Senel and Schütze, 2021) understanding of words using dictionary definitions. Again, matching multiple pieces of text with each other is much harder and therefore a promising task for benchmarking NLU.

5 Conclusion

We introduced CoDA21, a new challenging benchmark that tests natural language understanding capabilities of PLMs. Performing well on CoDA21 requires detailed understanding of contexts, performing complex inference and having world knowledge, which are crucial skills for NLP. All models we investigated perform clearly worse than humans, indicating a lack of these skills in the current state of the art in NLP. CoDA21 therefore is a promising benchmark for guiding the development of models with stronger NLU competence.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

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

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. Citeseer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

George A Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G Thomas. 1994. Using a semantic concordance for sense identification. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

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

A. Radford, Jeffrey Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. In *Technical Report*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

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

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

Timo Schick and Hinrich Schütze. 2020. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:8766–8774.

Lutfi Kerem Senel and Hinrich Schütze. 2021. Does she wink or does she nod? a challenging benchmark for evaluating word understanding of language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 532–538, Online. Association for Computational Linguistics.

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

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

A Appendices

A.1 CoDA21 group examples

Hidden word	**Context**
dust	1. He came spurring and whooping down the road , his horse kicking up clouds of <XXX> , shouting :
marble	2. Pels also sent a check for $ 100 to Russell 's widow and had a white <XXX> monument erected on his grave .
wastewater	3. The high cost of land and a few operational problems resulting from excessive loadings have created the need for a <XXX> treatment system with the operational characteristics of the oxidation pond but with the ability to treat more organic matter per unit volume .
feathers	4. It was a fine broody hen , white , with a maternal eye and a striking abundance of <XXX> in the under region of the abdomen .
fraction	5. It was then distilled at least three times from a trap at - 78 ‘ to a liquid air trap with only a small middle <XXX> being retained in each distillation .
soil	6. The thing is that these bees love a fine-grained <XXX> that is moist ; yet the water in the ground should not be stagnant either .
cards	7. And the coffee shop on Drexel Street , where the men spent their evenings and Sundays playing <XXX> , had a rose hedge beneath its window .
Synset	**Definition**
dust.n.01	1. fine powdery material such as dry earth or pollen that can be blown about in the air
marble.n.01	2. a hard crystalline metamorphic rock that takes a high polish; used for sculpture and as building material
effluent.n.01	3. water mixed with waste matter
feather.n.01	4. the light horny waterproof structure forming the external covering of birds
fraction.n.01	5. a component of a mixture that has been separated by a fractional process
soil.n.02	6. the part of the earth's surface consisting of humus and disintegrated rock
card.n.01	7. one of a set of small pieces of stiff paper marked in various ways and used for playing games or for telling fortunes

Table 5: A sample CoDA21 question taken from the noun part of the CoDA21-*clean-easy* dataset. The synsets are grandchildren of the parent synset ‘material.n.01’ whose definition is “the tangible substance that goes into the makeup of a physical object”.

Hidden word	**Context**
suggestion	1. This was Madden 's <XXX> ; the police chief shook his head over it .
concept	2. The <XXX> of apparent black-body temperature is used to describe the radiation received from the moon and the planets .
ideals	3. Religion can summate , epitomize , relate , and conserve all the highest <XXX> and values - ethical , aesthetic , and religious - of man formed in his culture .
reaction	4. That much of what he calls folklore is the result of beliefs carefully sown among the people with the conscious aim of producing a desired mass emotional <XXX> to a particular situation or set of situations is irrelevant .
feeling	5. He had an uneasy <XXX> about it .
programs	6. The Federal program of vocational education merely provides financial aid to encourage the establishment of vocational education <XXX> in public schools .
meaning	7. Indefinite reference also carries double <XXX> where an allusion to one person or thing seems to refer to another .
theme	8. Almost nothing is said of Charles ' spectacular victories , the central <XXX> being the heroic loyalty of the Swedish people to their idolized king in misfortune and defeat .
Synset	**Definition**
suggestion.n.01	1. an idea that is suggested
concept.n.01	2. an abstract or general idea inferred or derived from specific instances
ideal.n.01	3. the idea of something that is perfect; something that one hopes to attain
reaction.n.02	4. an idea evoked by some experience
impression.n.01	5. a vague idea in which some confidence is placed
plan.n.01	6. a series of steps to be carried out or goals to be accomplished
meaning.n.02	7. the idea that is intended
theme.n.02	8. a unifying idea that is a recurrent element in literary or artistic work

Table 6: A sample CoDA21 question taken from the noun part of the CoDA21-*clean-hard* dataset. The synsets are children of the parent synset 'idea.n.01' whose definition is "the content of cognition; the main thing you are thinking about".

On the Importance of Effectively Adapting Pretrained Language Models for Active Learning

Katerina Margatina♠ **Loïc Barrault**♣ **Nikolaos Aletras**♠
♠University of Sheffield, ♣University of Le Mans
{k.margatina,n.aletras}@sheffield.ac.uk
loic.barrault@univ-lemans.fr

Abstract

Recent Active Learning (AL) approaches in Natural Language Processing (NLP) proposed using off-the-shelf pretrained language models (LMs). In this paper, we argue that these LMs are not adapted effectively to the downstream task during AL and we explore ways to address this issue. We suggest to first adapt the pretrained LM to the target task by continuing training with all the available *unlabeled* data and then use it for AL. We also propose a simple yet effective fine-tuning method to ensure that the adapted LM is properly trained in both low and high resource scenarios during AL. Our experiments demonstrate that our approach provides substantial data efficiency improvements compared to the standard fine-tuning approach, suggesting that a poor training strategy can be catastrophic for AL.[1]

1 Introduction

Active Learning (AL) is a method for training supervised models in a data-efficient way (Cohn et al., 1996; Settles, 2009). It is especially useful in scenarios where a large pool of unlabeled data is available but only a limited annotation budget can be afforded; or where expert annotation is prohibitively expensive and time consuming. AL methods iteratively alternate between (i) model training with the labeled data available; and (ii) data selection for annotation using a stopping criterion, e.g. until exhausting a fixed annotation budget or reaching a pre-defined performance on a held-out dataset.

Data selection is performed by an acquisition function that ranks unlabeled data points by some *informativeness* metric aiming to improve over random selection, using either uncertainty (Lewis and Gale, 1994; Cohn et al., 1996; Gal et al., 2017; Kirsch et al., 2019; Zhang and Plank, 2021), diversity (Brinker, 2003; Bodó et al., 2011; Sener and Savarese, 2018), or both (Ducoffe and Precioso, 2018; Ash et al., 2020; Yuan et al., 2020; Margatina et al., 2021).

Previous AL approaches in NLP use task-specific neural models that are trained from scratch at each iteration (Shen et al., 2017; Siddhant and Lipton, 2018; Prabhu et al., 2019; Ikhwantri et al., 2018; Kasai et al., 2019). However, these models are usually outperformed by pretrained language models (LMs) adapted to end-tasks (Howard and Ruder, 2018), making them suboptimal for AL. Only recently, pretrained LMs such as BERT (Devlin et al., 2019) have been introduced in AL settings (Yuan et al., 2020; Ein-Dor et al., 2020; Shelmanov et al., 2021; Karamcheti et al., 2021; Margatina et al., 2021). Still, they are trained at each AL iteration with a standard fine-tuning approach that mainly includes a pre-defined number of training epochs, which has been demonstrated to be unstable, especially in small datasets (Zhang et al., 2020; Dodge et al., 2020; Mosbach et al., 2021). Since AL includes both low and high data resource settings, the AL model training scheme should be robust in both scenarios.[2]

To address these limitations, we introduce a suite of effective training strategies for AL (§2). Contrary to previous work (Yuan et al., 2020; Ein-Dor et al., 2020; Margatina et al., 2021) that also use BERT (Devlin et al., 2019), our proposed method accounts for various data availability settings and the instability of fine-tuning. First, we continue *pretraining* the LM with the available *unlabeled* data to adapt it to the task-specific domain. This way, we leverage not only the available labeled data at each AL iteration, but the entire unlabeled pool. Second, we further propose a simple yet effective fine-tuning method that is robust in both low and high resource data settings for AL.

[1] For all experiments in this paper, we have used the code provided by Margatina et al. (2021): `https://github.com/mourga/contrastive-active-learning`

[2] During the first few AL iterations the available labeled data is limited (*low-resource*), while it could become very large towards the last iterations (*high-resource*).

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 825 - 836
May 22-27, 2022

We explore the effectiveness of our approach on five standard natural language understandings tasks with various acquisition functions, showing that it outperforms all baselines (§3). We also conduct an analysis to demonstrate the importance of effective adaptation of pretrained models for AL (§4). Our findings highlight that the LM adaptation strategy can be more critical than the actual data acquisition strategy.

2 Adapting & Fine-tuning Pretrained Models for Active Learning

Given a downstream classification task with C classes, a typical AL setup consists of a pool of unlabeled data $\mathcal{D}_{\textbf{pool}}$, a model $\mathcal{M}$, an annotation budget b of data points and an acquisition function $a(.)$ for selecting k unlabeled data points for annotation (i.e. acquisition size) until b runs out. The AL performance is assessed by training a model on the actively acquired dataset and evaluating on a held-out test set $\mathcal{D}_{\textbf{test}}$.

Adaptation (TAPT) Inspired by recent work on transfer learning that shows improvements in downstream classification performance by continuing the pretraining of the LM with the task data (Howard and Ruder, 2018) we add an extra step to the AL process by continuing pretraining the LM (i.e. Task-Adaptive Pretraining TAPT), as in Gururangan et al. (2020). Formally, we use an LM, such as BERT (Devlin et al., 2019), $\mathcal{P}(x; W_0)$ with weights W_0, that has been already pretrained on a large corpus. We fine-tune $\mathcal{P}(x; W_0)$ with the available unlabeled data of the downstream task $\mathcal{D}_{\textbf{pool}}$, resulting in the task-adapted LM $\mathcal{P}_{\text{TAPT}}(x; W'_0)$ with new weights W'_0 (cf. line 2 of algorithm 1).

Fine-tuning (FT+) We now use the adapted LM $\mathcal{P}_{\text{TAPT}}(x; W'_0)$ for AL. At each iteration i, we initialize our model $\mathcal{M}_i$ with the pretrained weights W'_0 and we add a task-specific feedforward layer for classification with weights W_c on top of the `[CLS]` token representation of BERT-based $\mathcal{P}_{\text{TAPT}}$. We fine-tune the classification model $\mathcal{M}_i(x; [W'_0, W_c])$ with all $x \in \mathcal{D}_{\textbf{lab}}$. (cf. line 6 to 8 of algorithm 1).

Recent work in AL (Ein-Dor et al., 2020; Yuan et al., 2020) uses the standard fine-tuning method proposed in Devlin et al. (2019) which includes a fixed number of 3 training epochs, learning rate warmup over the first 10% of the steps and AdamW optimizer (Loshchilov and Hutter, 2019) without bias correction, among other hyperparameters.

Algorithm 1: AL with Pretrained LMs

```
Input: unlabeled data $\mathcal{D}_{\textbf{pool}}$, pretrained LM
       $\mathcal{P}(x; W_0)$, acquisition size $k$, AL
       iterations $T$, acquisition function $a$
1  $\mathcal{D}_{\textbf{lab}} \leftarrow \emptyset$
2  $\mathcal{P}_{\text{TAPT}}(x; W'_0) \leftarrow$ Train $\mathcal{P}(x; W_0)$ on $\mathcal{D}_{\textbf{pool}}$
3  $\mathcal{Q}_0 \leftarrow \text{RANDOM}(.), |\mathcal{Q}_0| = k$
4  $\mathcal{D}_{\textbf{lab}} = \mathcal{D}_{\textbf{lab}} \cup \mathcal{Q}_0$
5  $\mathcal{D}_{\textbf{pool}} = \mathcal{D}_{\textbf{pool}} \setminus \mathcal{Q}_0$
6  for $i \leftarrow 1$ to $T$ do
7      $\mathcal{M}_i(x; [W'_0, W_c]) \leftarrow$ Initialize from
         $\mathcal{P}_{\text{TAPT}}(x; W'_0)$
8      $\mathcal{M}_i(x; W_i) \leftarrow$ Train model on $\mathcal{D}_{\textbf{lab}}$
9      $\mathcal{Q}_i \leftarrow a(\mathcal{M}_i, \mathcal{D}_{\textbf{pool}}, k)$
10     $\mathcal{D}_{\textbf{lab}} = \mathcal{D}_{\textbf{lab}} \cup \mathcal{Q}_i$
11     $\mathcal{D}_{\textbf{pool}} = \mathcal{D}_{\textbf{pool}} \setminus \mathcal{Q}_i$
12 end
Output: $\mathcal{D}_{\textbf{lab}}$
```

We follow a different approach by taking into account insights from few-shot fine-tuning literature (Mosbach et al., 2021; Zhang et al., 2020; Dodge et al., 2020) that proposes longer fine-tuning and more evaluation steps during training. [3] We combine these guidelines to our fine-tuning approach by using early stopping with 20 epochs based on the validation loss, learning rate $2e-5$, bias correction and 5 evaluation steps per epoch. However, increasing the number of epochs from 3 to 20, also increases the warmup steps (10% of total steps[4]) almost 7 times. This may be problematic in scenarios where the dataset is large but the optimal number of epochs may be small (e.g. 2 or 3). To account for this limitation in our AL setting where the size of training set changes at each iteration, we propose to select the warmup steps as $min(10\%$ of total steps$, 100)$. We denote standard fine-tuning as SFT and our approach as FT+.

3 Experiments & Results

Data We experiment with five diverse natural language understanding tasks: question classification

[3] In this paper we use *few-shot* to describe the setting where there are *few* labeled data available and therefore *few-shot fine-tuning* corresponds to fine-tuning a model on limited labeled training data. This is different than the few-shot setting presented in recent literature (Brown et al., 2020), where no model weights are updated.

[4] Some guidelines propose an even smaller number of warmup steps, such as 6% in RoBERTa (Liu et al., 2020).

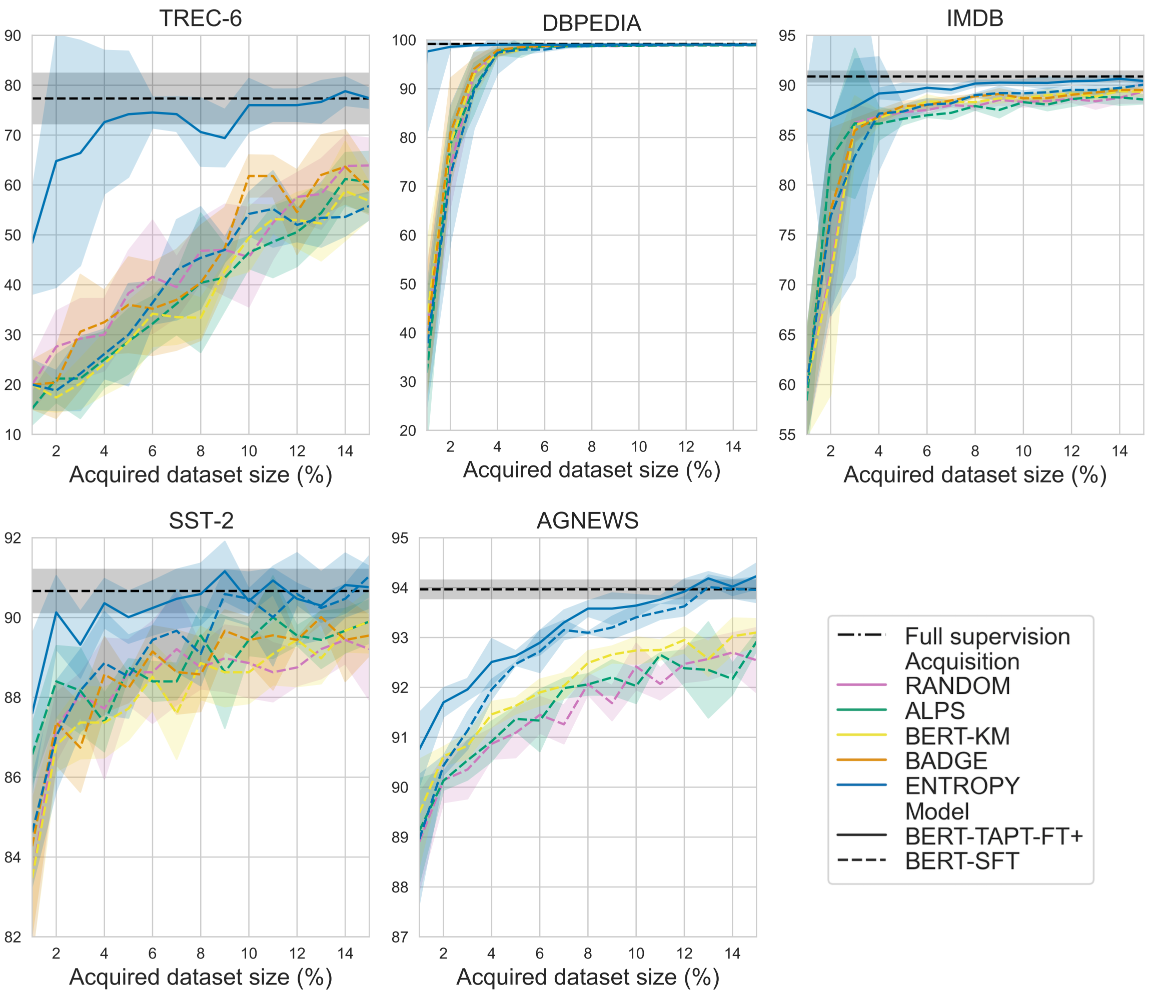


Figure 1: Test accuracy during AL iterations. We plot the median and standard deviation across five runs.

DATASETS	TRAIN	VAL	TEST	k	C
TREC-6	4.9K	546	500	1%	6
DBPEDIA	20K	2K	70K	1%	14
IMDB	22.5K	2.5K	25K	1%	2
SST-2	60.6K	6.7K	871	1%	2
AGNEWS	114K	6K	7.6K	0.5%	4

Table 1: Datasets statistics for $\mathcal{D}_{\textbf{pool}}$, $\mathcal{D}_{\textbf{val}}$ and $\mathcal{D}_{\textbf{test}}$ respectively. k stands for the acquisition size (% of $\mathcal{D}_{\textbf{pool}}$) and C the number of classes.

(TREC-6; Voorhees and Tice (2000)), sentiment analysis (IMDB; Maas et al. (2011), SST-2 Socher et al. (2013)) and topic classification (DBPEDIA, AGNEWS; Zhang et al. (2015)), including binary and multi-class labels and varying dataset sizes (Table 1). More details can be found in Appendix A.1.

Experimental Setup We perform all AL experiments using `BERT-base` (Devlin et al., 2019) and ENTROPY, BERTKM, ALPS (Yuan et al., 2020), BADGE (Ash et al., 2020) and RANDOM (baseline) as the acquisition functions. We pair our proposed training approach TAPT-FT+ with ENTROPY acquisition. We refer the reader to Appendix A for an extended description of our experimental setup, including the datasets used (§A.1), the training and AL details (§A.2), the model hyperparameters (§A.3) and the baselines (§A.4).

Results Figure 1 shows the test accuracy during AL iterations. We first observe that our proposed approach (TAPT-FT+) achieves large data efficiency reaching the full-dataset performance within the 15% budget for all datasets, in contrast to the standard AL approach (BERT-SFT). The effectiveness of our approach is mostly notable in the smaller datasets. In TREC-6, it achieves the goal accuracy with almost 10% annotated data, while in DBPEDIA only in the first iteration with 2% of the data. After the first AL iteration in IMDB, TAPT-FT+, it achieves only 2.5 points of accuracy lower than the performance when using 100% of the data. In the larger SST-2 and AGNEWS datasets, it is closer to the baselines but still outperforms them, achieving the full-dataset performance with 8% and 12% of the data respectively. We also observe that in all five datasets, the addition of our proposed pretraining step (TAPT) and fine-tuning technique (FT+) leads to large performance gains, especially in the first AL iterations. This is particularly evident in TREC-6, DBPEDIA and IMDB datasets, where after the *first* AL iteration (i.e. equivalent to 2% of training data) TAPT+FT+ with ENTROPY is 45, 30 and 12 points in accuracy higher than the ENTROPY baseline with BERT and SFT.

Training vs. Acquisition Strategy We finally observe that the performance curves of the various acquisition functions considered (i.e. dotted lines) are generally close to each other, suggesting that the choice of the acquisition strategy may not affect substantially the AL performance in certain cases. In other words, we conclude that *the training strategy can be more important than the acquisition strategy*. We find that uncertainty sampling with ENTROPY is generally the best performing acquisition function, followed by BADGE.[5] Still, finding a universally well-performing acquisition function, independent of the training strategy, is an open research question.

4 Analysis & Discussion

4.1 Task-Adaptive Pretraining

We first present details of our implementation of TAPT (§2) and reflect on its effectiveness in the AL pipeline. Following Gururangan et al. (2020), we continue pretraining BERT for the MLM task using all the unlabeled data $\mathcal{D}_{\textbf{pool}}$ for all datasets separately. We plot the learning curves of BERT-TAPT for all datasets in Figure 2. We first observe that the masked LM loss is steadily decreasing for DBPEDIA, IMDB and AGNEWS across optimization steps, which correlates with the high early AL performance gains of TAPT in these datasets (Fig. 1). We also observe that the LM overfits in TREC-6 and SST-2 datasets. We attribute this to the very small training dataset of TREC-6 and the informal textual style of SST-2. Despite the fact that the SST-2 dataset includes approximately 67K of training data, the sentences are very short (i.e. average length of 9.4 words per sentence). We hypothesize the LM overfits because of the lack of long and more diverse sentences. We provide more details on TAPT at the Appendix B.1.

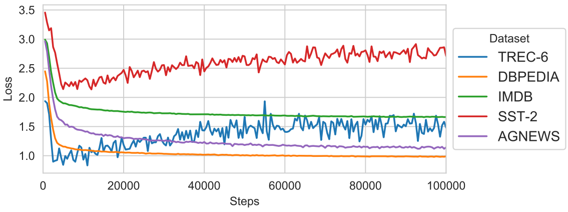


Figure 2: Validation MLM loss during TAPT.

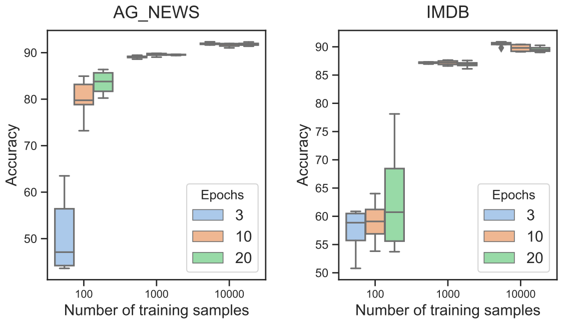


Figure 3: Few-shot standard BERT fine-tuning.

4.2 Few-shot Fine-tuning

In this set of experiments, we aim to highlight that it is crucial to consider the few-shot learning problem in the early AL stages, which is often neglected in literature. This is more important when using pretrained LMs, since they are overparameterized models that require adapting their training scheme in low data settings to ensure robustness.

To illustrate the potential ineffectiveness of standard fine-tuning (SFT), we randomly undersample the AGNEWS and IMDB datasets to form low, medium and high resource data settings (i.e. 100, 1,000 and 10,000 training samples), and train BERT for a fixed number of 3, 10, and 20 epochs. We repeat this process with 10 different random seeds to account for stochasticity in sampling and we plot the test accuracy in Figure 3. Figure 3 shows that SFT is suboptimal for low data settings (e.g. 100 samples), indicating that more optimization steps (i.e. epochs) are needed for the model to adapt to the few training samples (Zhang et al., 2020; Mosbach et al., 2021). As the training samples increase (e.g. 1,000), fewer epochs are often better. It is thus evident that there is not a clearly optimal way to choose a predefined number

[5] We provide results with additional acquisition functions in the Appendix B.2 and B.3.

of epochs to train the model given the number of training examples. This motivates the need to find a fine-tuning policy for AL that effectively adapts to the data resource setting of each iteration (independently of the number of training examples or dataset), which is mainly tackled by our proposed fine-tuning approach FT+ (§2).

4.3 Ablation Study

We finally conduct an ablation study to evaluate the contribution of our two proposed steps to the AL pipeline; the pretraining step (TAPT) and fine-tuning method (FT+). We show that the addition of both methods provides large gains compared to standard fine-tuning (SFT) in terms of accuracy, data efficiency and uncertainty calibration. We compare BERT with SFT, BERT with FT+ and BERT-TAPT with FT+. Along with test accuracy, we also evaluate each model using uncertainty estimation metrics (Ovadia et al., 2019): Brier score, negative log likelihood (NLL), expected calibration error (ECE) and entropy. A well-calibrated model should have high accuracy and low uncertainty.

Figure 4 shows the results for the smallest and largest datasets, TREC-6 and AGNEWS respectively. For TREC-6, training BERT with our fine-tuning approach FT+ provides large gains both in accuracy and uncertainty calibration, showing the importance of fine-tuning the LM for a larger number of epochs in low resource settings. For the larger dataset, AGNEWS, we see that BERT with SFT performs equally to FT+ which is the ideal scenario. We see that our fine-tuning approach does not deteriorate the performance of BERT given the large increase in warmup steps, showing that our simple strategy provides robust results in both high and low resource settings. After demonstrating that FT+ yields better results than SFT, we next compare BERT-TAPT-FT+ against BERT-FT+. We observe that in both datasets BERT-TAPT outperforms BERT, with this being particularly evident in the early iterations. This confirms our hypothesis that by implicitly using the entire pool of unlabeled data for extra pretraining (TAPT), we boost the performance of the AL model using less data.

5 Conclusion

We have presented a simple yet effective training scheme for AL with pretrained LMs that accounts for varying data availability and instability of fine-tuning. Specifically, we propose to first continue pretraining the LM with the available unlabeled data to *adapt* it to the task-specific domain. This way, we leverage not only the available labeled data at each AL iteration, but the entire unlabeled pool. We further propose a method to *fine-tune* the model during AL iterations so that training is robust in both low and high resource data settings.

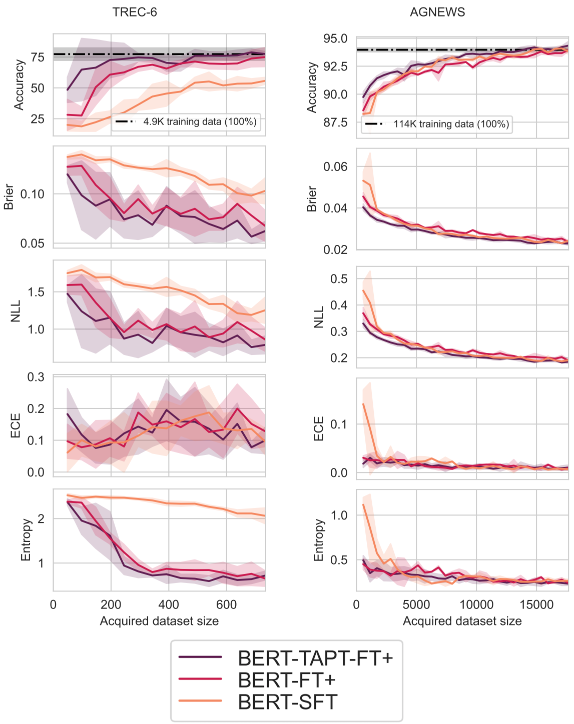


Figure 4: Ablation study for TAPT and FT+.

Our experiments show that our approach yields substantially better results than standard fine-tuning in five standard NLP datasets. Furthermore, we find that *the training strategy can be more important than the acquisition strategy*. In other words, a poor training strategy can be a crucial impediment to the effectiveness of a good acquisition function, and thus limit its effectiveness (even over random sampling). Hence, our work highlights how critical it is to properly adapt a pretrained LM to the low data resource AL setting.

As state-of-the-art models in NLP advance rapidly, in the future we would be interested in exploring the use of larger LMs, such as GPT-3 (Brown et al., 2020) and FLAN (Wei et al., 2022). These models have achieved impressive performance in very low data resource settings (e.g. zero-shot and few-shot), so we would imagine they would be good candidates for the challenging setting of active learning.

Acknowledgments

We would like to thank Giorgos Vernikos, our colleagues at the Sheffield NLP group for feedback on an earlier version of this paper, and all the anonymous reviewers for their constructive comments. KM and NA are supported by Amazon through the Alexa Fellowship scheme.

References

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2020. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*.

Zalán Bodó, Zsolt Minier, and Lehel Csató. 2011. Active learning with clustering. In *Proceedings of the Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, volume 16, pages 127–139.

Klaus Brinker. 2003. Incorporating diversity in active learning with support vector machines. In *Proceedings of the International Conference on Machine Learning*, pages 59–66.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4(1):129–145.

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

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *ArXiv*.

Melanie Ducoffe and Frederic Precioso. 2018. Adversarial active learning for deep networks: a margin based approach.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active learning for BERT: An empirical study. In *Proceedings of theConference on Empirical Methods in Natural Language Processing*, pages 7949–7962.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the International Conference on Machine Learning*, volume 48, pages 1050–1059.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep Bayesian active learning with image data. In *Proceedings of the International Conference on Machine Learning*, volume 70, pages 1183–1192.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. 2011. Bayesian active learning for classification and preference learning. *ArXiv*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 328–339.

Fariz Ikhwantri, Samuel Louvan, Kemal Kurniawan, Bagas Abisena, Valdi Rachman, Alfan Farizki Wicaksono, and Rahmad Mahendra. 2018. Multi-task active learning for neural semantic role labeling on low resource conversational corpus. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 43–50.

Siddharth Karamcheti, Ranjay Krishna, Li Fei-Fei, and Christopher Manning. 2021. Mind your outliers! investigating the negative impact of outliers on active learning for visual question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7265–7281, Online. Association for Computational Linguistics.

Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. 2019. Low-resource deep entity resolution with transfer and active learning. In *Proceedings of the Conference of the Association for Computational Linguistic*, pages 5851–5861.

Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. 2019. BatchBALD: Efficient and diverse batch acquisition for deep bayesian active learning. In *Neural Information Processing Systems*, pages 7026–7037.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *In Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

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

David Lowell and Zachary C Lipton. 2019. Practical obstacles to deploying active learning. *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 21–30.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 650–663, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning {bert}: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, volume 32, pages 13991–14002.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Ameya Prabhu, Charles Dognin, and Maneesh Singh. 2019. Sampling bias in deep active classification: An empirical study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 4056–4066.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*.

Burr Settles. 2009. Active learning literature survey. Computer sciences technical report.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*.

Artem Shelmanov, Dmitri Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry V. Dylov, and Alexander Panchenko. 2021. Active learning for sequence tagging with deep pre-trained models and Bayesian uncertainty estimates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1698–1712, Online. Association for Computational Linguistics.

Yanyao Shen, Hyokun Yun, Zachary Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep active learning for named entity recognition. In *Proceedings of the Workshop on Representation Learning for NLP*, pages 252–256.

Aditya Siddhant and Zachary C Lipton. 2018. Deep bayesian active learning for natural language processing: Results of a Large-Scale empirical study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

N Srivastava, G Hinton, A Krizhevsky, and others. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Ellen Voorhees and Dawn Tice. 2000. The trec-8 question answering track evaluation. *Proceedings of the Text Retrieval Conference*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling.

Mike Zhang and Barbara Plank. 2021. Cartography active learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 395–406, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2020. Revisiting few-sample bert fine-tuning. *ArXiv*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, pages 649–657. Curran Associates, Inc.

A Appendix: Experimental Setup

A.1 Datasets

We experiment with five diverse natural language understanding tasks including binary and multi-class labels and varying dataset sizes (Table 1). The first task is question classification using the six-class version of the small TREC-6 dataset of open-domain, fact-based questions divided into broad semantic categories (Voorhees and Tice, 2000). We also evaluate our approach on sentiment analysis using the binary movie review IMDB dataset (Maas et al., 2011) and the binary version of the SST-2 dataset (Socher et al., 2013). We finally use the large-scale AGNEWS and DBPEDIA datasets from Zhang et al. (2015) for topic classification. We undersample the latter and form a $\mathcal{D}_{\textbf{pool}}$ of 20K examples and $\mathcal{D}_{\textbf{val}}$ 2K as in Margatina et al. (2021). For TREC-6, IMDB and SST-2 we randomly sample 10% from the training set to serve as the validation set, while for AGNEWS we sample 5%. For the DBPEDIA dataset we undersample both training and validation datasets (from the standard splits) to facilitate our AL simulation (i.e. the original dataset consists of 560K training and 28K validation data examples). For all datasets we use the standard test set, apart from the SST-2 dataset that is taken from the GLUE benchmark (Wang et al., 2019) we use the development set as the held-out test set (and subsample a development set from the original training set).

A.2 Training & AL Details

We use BERT-BASE (Devlin et al., 2019) and fine-tune it (TAPT §2) for 100K steps, with learning rate $2e-05$ and the rest of hyperparameters as in Gururangan et al. (2020) using the HuggingFace library (Wolf et al., 2020). We evaluate the model 5 times per epoch on $\mathcal{D}_{val}$ and keep the one with the lowest validation loss as in Dodge et al. (2020). We use the code provided by Kirsch et al. (2019) for the uncertainty-based acquisition functions and Yuan et al. (2020) for ALPS, BADGE and BERTKM. We use the standard splits provided for all datasets, if available, otherwise we randomly sample a validation set. We test all models on a held-out test set. We repeat all experiments with five different random seeds resulting into different initializations of $\mathcal{D}_{\textbf{lab}}$ and the weights of the extra task-specific output feedforward layer. For all datasets we use as budget the 15% of $\mathcal{D}_{\textbf{pool}}$. Each experiment is run on a single Nvidia Tesla V100 GPU.

A.3 Hyperparameters

For all datasets we train BERT-BASE (Devlin et al., 2019) from the HuggingFace library (Wolf et al., 2020) in Pytorch (Paszke et al., 2019). We train all models with batch size 16, learning rate $2e-5$, no weight decay, AdamW optimizer with epsilon $1e-8$. For all datasets we use maximum sequence length of 128, except for IMDB and AGNEWS that contain longer input texts, where we use 256. To ensure reproducibility and fair comparison between the various methods under evaluation, we run all experiments with the same five seeds that we randomly selected from the range $[1, 9999]$.

A.4 Baselines

Acquisition functions We compare ENTROPYwith four baseline acquisition functions. The first is the standard AL baseline, **RANDOM**, which applies uniform sampling and selects k data points from $\mathcal{D}_{\textbf{pool}}$ at each iteration. The second is **BADGE** (Ash et al., 2020), an acquisition function that aims to combine diversity and uncertainty sampling. The algorithm computes *gradient embeddings* g_x for every candidate data point x in $\mathcal{D}_{\textbf{pool}}$ and then uses clustering to select a batch. Each g_x is computed as the gradient of the cross-entropy loss with respect to the parameters of the model's last layer. We also compare against a recently introduced cold-start acquisition function called **ALPS** (Yuan et al., 2020). ALPS acquisition uses the masked language model (MLM) loss of BERT as a proxy for model uncertainty in the downstream classification task. Specifically, aiming to leverage both uncertainty and diversity, ALPS forms a *surprisal embedding* s_x for each x, by passing the unmasked input x through the BERT MLM head to compute the cross-entropy loss for a random 15% subsample of tokens against the target labels. ALPS clusters these embeddings to sample k sentences for each AL iteration. Last, following Yuan et al. (2020), we use **BERTKM** as a diversity baseline, where the l_2 normalized BERT output embeddings are used for clustering.

Models & Fine-tuning Methods We evaluate two variants of the pretrained language model; the original **BERT** model, used in Yuan et al. (2020) and Ein-Dor et al. (2020)[6], and our adapted model **BERT-TAPT** (§2), and two fine-tuning methods;

[6] Ein-Dor et al. (2020) evaluate various acquisition functions, including entropy with MC dropout, and use BERT with the standard fine-tuning approach (SFT).

our proposed fine-tuning approach **FT+** (§2) and standard BERT fine-tuning **SFT**.

MODEL	TREC-6	DBPEDIA	IMDB	SST-2	AGNEWS
		VALIDATION SET			
BERT	94.4	99.1	90.7	93.7	94.4
BERT-TAPT	95.2	99.2	91.9	94.3	94.5
		TEST SET			
BERT	80.6	99.2	91.0	90.6	94.0
BERT-TAPT	77.2	99.2	91.9	90.8	94.2

Table 2: Accuracy with 100% of data over five runs (different random seeds).

B Appendix: Analysis

B.1 Task-Adaptive Pretraining (TAPT) & Full-Dataset Performance

As discussed in §2 and §4, we continue training the BERT-BASE (Devlin et al., 2019) pretrained masked language model using the available data $\mathcal{D}_{\textbf{pool}}$. We explored various learning rates between $1e-4$ and $1e-5$ and found the latter to produce the lowest validation loss. We trained each model (one for each dataset) for up to 100K optimization steps, we evaluated on $\mathcal{D}_{\textbf{val}}$ every 500 steps and saved the checkpoint with the lowest validation loss. We used the resulting model in our (BERT-TAPT) experiments. We plot the learning curves of masked language modeling task (TAPT) for three datasets and all considered learning rates in Figure 5. We notice that a smaller learning rate facilitates the training of the MLM.

In Table 2 we provide the validation and test accuracy of BERT and BERT-TAPT for all datasets. We present the mean across runs with three random seeds. For fine-tuning the models, we used the proposed approach FT+ (§2).

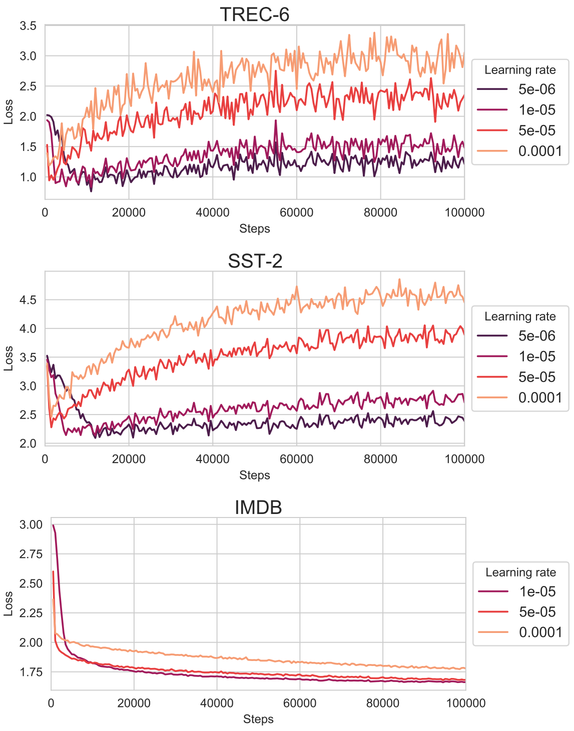


Figure 5: Learning curves of TAPT for various learning rates.

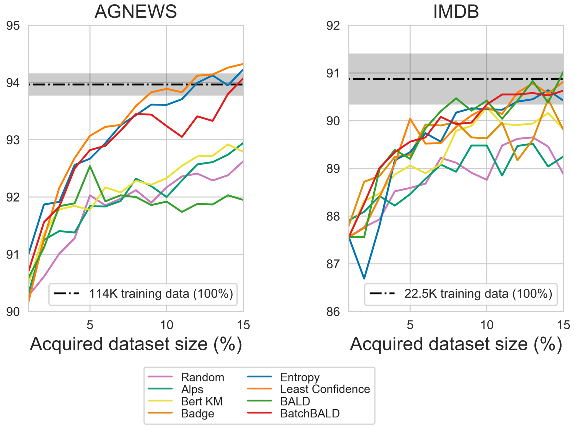


Figure 6: Comparison of acquisition functions using TAPT and FT+ in training BERT.

B.2 Performance of Acquisition Functions

In our BERT-TAPT-FT+ experiments so far, we showed results with ENTROPY. We have also experimented with various uncertainty-based acquisition functions. Specifically, four uncertainty-based acquisition functions are used in our work: LEAST CONFIDENCE, ENTROPY, BALD and BATCHBALD. LEAST CONFIDENCE (Lewis and Gale, 1994) sorts $\mathcal{D}_{pool}$ by the probability of *not* predicting the most confident class, in descending order, ENTROPY (Shannon, 1948) selects samples that maximize the predictive entropy, and BALD (Houlsby et al., 2011), short for Bayesian Active Learning by Disagreement, chooses data points that maximize the mutual information between predictions and model's posterior probabilities. BATCHBALD (Kirsch et al., 2019) is a recently introduced extension of BALD that *jointly* scores points by estimating the mutual information between multiple data points and the model parameters. This iterative algorithm aims to find *batches* of informative data points, in contrast to BALD that chooses points that are informative individually. Note that LEAST CONFIDENCE, ENTROPY and BALD have been used in AL for NLP by Siddhant and Lipton (2018). To the best of our

	TREC-6	SST-2	IMDB	DBPEDIA	AGNEWS
RANDOM	0/0	0/0	0/0	0/0	0/0
ALPS	0/57	0/478	0/206	0/134	0/634
BADGE	0/63	0/23110	0/1059	0/192	-
BERTKM	0/47	0/2297	0/324	0/137	0/3651
ENTROPY	81/0	989/0	557/0	264/0	2911/0
LEAST CONFIDENCE	69/0	865/0	522/0	256/0	2607/0
BALD	69/0	797/0	524/0	256/0	2589/0
BATCHBALD	69/21	841/1141	450/104	256/482	2844/5611

Table 3: Runtimes (in seconds) for all datasets. In each cell of the table we present a tuple i/s where i is the *inference time* and s the *selection time*. *Inference time* is the time for the model to perform a forward pass for all the unlabeled data in $\mathcal{D}_{\textbf{pool}}$ and *selection time* is the time that each acquisition function requires to rank all candidate data points and select k for annotation (for a single iteration). Since we cannot report the runtimes for every model in the AL pipeline (at each iteration the size of $\mathcal{D}_{\textbf{pool}}$ changes), we provide the median.

knowledge, BATCHBALD is evaluated for the first time in the NLP domain.

Instead of using the output softmax probabilities for each class, we use a probabilistic formulation of deep neural networks in order to acquire better calibrated scores. Monte Carlo (MC) dropout (Gal and Ghahramani, 2016) is a simple yet effective method for performing approximate variational inference, based on dropout (Srivastava et al., 2014). Gal and Ghahramani (2016) prove that by simply performing *dropout during the forward pass in making predictions*, the output is equivalent to the prediction when the parameters are sampled from a variational distribution of the true posterior. Therefore, dropout during inference results into obtaining predictions from different parts of the network. Our BERT-based $\mathcal{M}_i$ model uses dropout layers during training for regularization. We apply MC dropout by simply activating them during test time and we perform multiple stochastic forward passes. Formally, we do N passes of every $x \in \mathcal{D}_{\textbf{pool}}$ through $\mathcal{M}_i(x; W_i)$ to acquire N different output probability distributions for each x. MC dropout for AL has been previously used in the literature (Gal et al., 2017; Shen et al., 2017; Siddhant and Lipton, 2018; Lowell and Lipton, 2019; Ein-Dor et al., 2020; Shelmanov et al., 2021).

Our findings show that all functions provide similar performance, except for BALD that slightly underperforms. This makes our approach agnostic to the selected uncertainty-based acquisition method. We also evaluate our proposed methods with our baseline acquisition functions, i.e. RANDOM, ALPS, BERTKM and BADGE, since our training strategy is orthogonal to the acquisition strategy. We compare all acquisition functions with BERT-TAPT-FT+ for AGNEWS and IMDB in Figure 6. We observe that in general uncertainty-based acquisition performs better compared to diversity, while all acquisition strategies have benefited from our training strategy (TAPT and FT+).

B.3 Efficiency of Acquisition Functions

In this section we discuss the efficiency of the eight acquisition functions considered in this work; RANDOM, ALPS, BADGE, BERTKM, ENTROPY, LEAST CONFIDENCE, BALD and BATCHBALD.

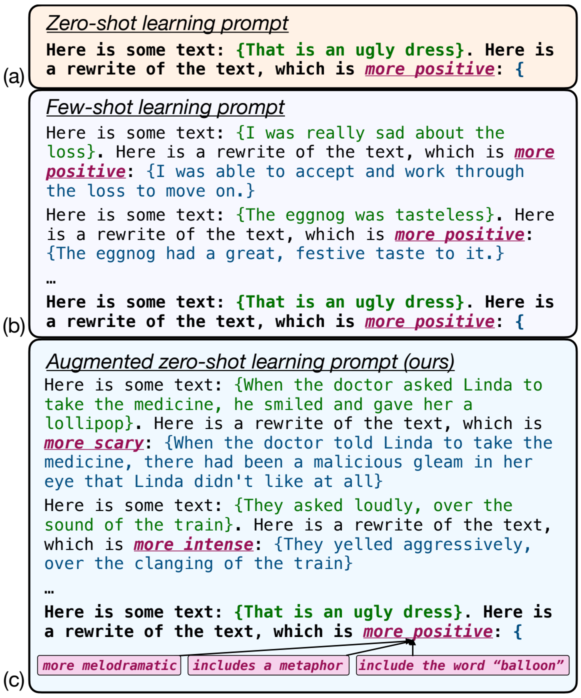
In Table 3 we provide the runtimes for all acquisition functions and datasets. Each AL experiments consists of multiple iterations and (therefore multiple models), each with a different training dataset $\mathcal{D}_{\textbf{lab}}$ and pool of unlabeled data $\mathcal{D}_{\textbf{pool}}$. In order to evaluate how computationally heavy is each method, we provide the *median* of all the models in one AL experiment. We calculate the runtime of two types of functionalities. The first is the *inference time* and stands for the forward pass of each $x \in \mathcal{D}_{\textbf{pool}}$ to acquire confidence scores for uncertainty sampling. RANDOM, ALPS, BADGE and BERTKM do not require this step so it is only applied of uncertainty-based acquisition where acquiring uncertainty estimates with MC dropout is needed. The second functionality is *selection time* and measures how much time each acquisition function requires to rank and select the k data points from $\mathcal{D}_{\textbf{pool}}$ to be labeled in the next step of the AL pipeline. RANDOM, ENTROPY, LEAST CONFIDENCE and BALD perform simple equations to rank the data points and therefore so do not require selection time. On the other hand, ALPS, BADGE, BERTKM and BATCHBALD perform iterative algorithms that increase selection time. From all acquisition functions ALPS and BERTKM are faster because they do not require the inference step of all the unlabeled data to the model. ENTROPY, LEAST CONFIDENCE and BALD require the same time for selecting data, which is equivalent for the time needed to perform one forward pass of the entire $\mathcal{D}_{\textbf{pool}}$. Finally BADGE and BATCHBALD are the most computationally heavy approaches, since both algorithms require multiple computations for the *selection time*. RANDOM has a total runtime of zero seconds, as expected.

A Recipe For Arbitrary Text Style Transfer with Large Language Models

Emily Reif[1*] **Daphne Ippolito**[1,2*] **Ann Yuan**[1] **Andy Coenen**[1]
Chris Callison-Burch[2] **Jason Wei**[1]
[1]Google Research [2]University of Pennsylvania
{ereif, annyuan, andycoenen, jasonwei}@google.com
{daphnei, ccb}@seas.upenn.edu

Abstract

In this paper, we leverage large language models (LMs) to perform zero-shot text style transfer. We present a prompting method that we call *augmented zero-shot learning*, which frames style transfer as a sentence rewriting task and requires only a natural language instruction, without model fine-tuning or exemplars in the target style. Augmented zero-shot learning is simple and demonstrates promising results not just on standard style transfer tasks such as sentiment, but also on natural language transformations such as "make this melodramatic" or "insert a metaphor."



Zero-shot learning prompt

Here is some text: {That is an ugly dress}. Here is a rewrite of the text, which is *more positive*: {

Few-shot learning prompt

Here is some text: {I was really sad about the loss}. Here is a rewrite of the text, which is *more positive*: {I was able to accept and work through the loss to move on.}

Here is some text: {The eggnog was tasteless}. Here is a rewrite of the text, which is *more positive*: {The eggnog had a great, festive taste to it.}

…

Here is some text: {That is an ugly dress}. Here is a rewrite of the text, which is *more positive*: {

Augmented zero-shot learning prompt (ours)

Here is some text: {When the doctor asked Linda to take the medicine, he smiled and gave her a lollipop}. Here is a rewrite of the text, which is *more scary*: {When the doctor told Linda to take the medicine, there had been a malicious gleam in her eye that Linda didn't like at all}

Here is some text: {They asked loudly, over the sound of the train}. Here is a rewrite of the text, which is *more intense*: {They yelled aggressively, over the clanging of the train}

…

Here is some text: {That is an ugly dress}. Here is a rewrite of the text, which is *more positive*: {

Figure 1: Zero-shot, few-shot, and augmented zero-shot prompts for style transfer. The boldface text is the zero-shot prompt, and the plain text is the additional priming sequence. The full prompts used in this paper are shown in Table 7. We encourage readers to examine the outputs of our model at https://bit.ly/3fLDuci.

1 Introduction

Text style transfer is the task of rewriting text to incorporate additional or alternative stylistic elements while preserving the overall semantics and structure. Although style transfer has garnered increased interest due to the success of deep learning, these approaches usually require a substantial amount of labeled training examples, either as parallel text data (Zhu et al., 2010; Rao and Tetreault, 2018) or non-parallel text data of a single style. (Li et al., 2018; Jin et al., 2019; Liu et al., 2020; Krishna et al., 2020). Even bleeding-edge approaches that tackle the challenging problem of label-free style transfer are limited in that they require at least several exemplar sentences that dictate a given target style (Xu et al., 2020; Riley et al., 2021). Hence, recent survey papers have identified a need for new methods that both reduce the training data requirements and expand the scope of styles supported (Jin et al., 2020; Hu et al., 2020).

In this work, we present *augmented zero-shot learning*, a prompting method that allows large language models to perform text style transfer to arbitrary styles, without any exemplars in the target style. Our method builds on prior work showing that sufficiently large LMs such as GPT-3 can perform various tasks ranging from classification to translation, simply by choosing a clever prompt to prepend to the input text for which the model is asked to continue (Brown et al., 2020; Branwen, 2020). Using a single prompt that provides several demonstrations of sentences being "rewritten" to meet a desired condition, language models can extrapolate and rewrite text in unseen styles. We are thus able to perform style transfer to arbitrary styles such as "*make this sentence more comic*" or "*include the word balloon*."

Augmented zero-shot learning is simple and facilitates the application of style transfer to a wider

[*] Equal contribution

range of styles than existing work. Our contributions are the following.

1. We propose a recipe for style transfer using large LMs that is label-free, training-free, and intuitively controllable.
2. Via human evaluation, we find that our method achieves strong performance on both standard and non-standard style transfer tasks. We also compare our approach for sentiment transfer with prior methods using automatic evaluation.
3. We explore real-world desired style transfers generated from users of a text editing UI that implements our method.

2 Augmented zero-shot prompting

Although large LMs are trained only for continuation, recent work has shown that they can perform a variety of NLP tasks by expressing the task as a prompt that encourages the model to output the desired answer as the continuation (Puri and Catanzaro, 2019; Weller et al., 2020; Brown et al., 2020; Schick and Schütze, 2021, *inter alia*; see Liu et al. (2021a) for a survey). The simplest approach, **zero-shot prompting**, directly uses natural language to ask the large LM to perform a task, as shown in Figure 1a. Zero-shot prompting, however, can be prone to failure modes such as not returning well-formatted or logical outputs (see §6). **Few-shot prompting**, as shown in Figure 1b, has been shown to achieve higher performance, but requires exemplars for the exact task that we want the model to perform. Such few-shot examples can be easily obtained if the desired style transformation is known ahead of time, but this ultimately limits style transfer to a set of pre-specified style tasks.

To remove the need for these labeled exemplars for each style transfer task, we propose *augmented zero-shot learning*, a method for performing multi-task style transfer using a single set of exemplars. Instead of prompting the model with exemplars specific to the exact style transfer task we wish to perform, we prompt the model with examples of a variety of sentence rewriting operations, as shown in Figure 1c. This intuition is inspired by Reynolds and McDonell (2021)'s observation that successful prompts constrain the behavior of the large LM away from failure modes—in our case, we aim to preserve the flexibility of a zero shot prompt while encouraging the model to produce outputs of a specific template. We keep the the format of the exemplars constant and insert the desired sentence transformation into the same format. In this way, the augmented zero-shot formulation supports arbitrary sentence rewriting tasks without the need to write any task-specific exemplars. Thus, it works for a wide range of styles, including modifying the text to be "*more melodramatic,*" "*insert a metaphor,*" or "*include the word balloon.*"

Original	`There, in the middle of the street, stood an old man in a weatherbeaten brown coat.`
More descriptive	`The man's brown trenchcoat was coated with snow, and as he leaned against the telephone booth, he cast a long shadow across the road in the late afternoon sun.`
Include a metaphor	`There, in the middle of the street, stood an old man whose face resembled the weatherworn bark of a gnarled tree.`
Include the word "*park*"	`There, in the middle of Central Park, stood an old man in a weatherbeaten brown coat.`
Include the word "*balloon*"	`There, in the middle of the street, stood an old man with several colourful balloons tied to the straps of his coat.`
More comic	`There, in the middle of the street, stood an old man in a raggedy brown coat that was much too big for him.`
More melodramatic	`There, in the middle of the road, stood a grizzled old man, the light of life faded from his sunken eyes.`

Table 1: Example style transfer outputs from augmented zero-shot learning for non-standard styles.

3 Experimental Setup

Style transfer tasks. We consider six style transfer tasks that we deem non-standard, listed in Table 1. These styles were chosen to be representative of most frequent style adjustments made by users of an AI-assisted text editor that employs our method (discussed further in §5). As source sentences, we use 50 sentences randomly drawn from the Reddit Writing Prompts validation set (Fan et al., 2018), excluding those that already clearly exhibited one of the styles or were ungrammatical/incoherent. We use human evaluation for these styles, since not all styles have readily available classifiers.

We also evaluate our method on two standard style transfer tasks: sentiment and formality. We use the Yelp polarity dataset (Zhang et al., 2015) for sentiment and Grammarly's Yahoo Answers Formality Corpus (GYAFC) dataset for formality (Rao and Tetreault, 2018).[1] These datasets allow us to evaluate performance of augmented zero-shot learning in the context of prior supervised methods which have been used on these tasks.

[1] Hosted by Luo et al. (2019).

Model. Augmented zero-shot learning requires a large language model. We primarily use LaMDA, a left-to-right decoder-only transformer language model (Vaswani et al., 2017) with a non-embedding parameter count of 137B (Thoppilan et al., 2022). The pre-trained LaMDA model, which we refer to as *LLM*, was trained on a corpus comprising 1.95B public web documents, including forum and dialog data and Wikipedia. The dataset was tokenized into 2.49T BPE tokens with a SentencePiece vocabulary size of 32K (Kudo and Richardson, 2018). We also use *LLM-Dialog*, the final LaMDA model which was finetuned on a curated, high-quality subset of data identified to be in a conversational format. Decoding was done with top-k=40. To show that the success of augmented zero-shot learning is not restricted to these two large LMs, we also perform experiments with GPT-3 (Table 8). For GPT-3, decoding was done with nucleus sampling using p=0.6 (Holtzman et al., 2019).

The prompts used for *LLM* and GPT-3 are shown in Figure 1. For *LLM-Dialog*, the prompt was instead formulated as a conversation between one agent who is requesting rewrites and another who is performing the rewrites. See Table 7 in the Appendix for the full non-abbreviated prompts.

4 Results

4.1 Non-Standard Styles

For our six non-standard styles, we asked six professional raters to assess <input sentence, target style, output sentence> tuples. These raters are fluent in English, live in India, and work full time labeling and evaluating data. To decrease inter-rater discrepancy and ensure that our instructions were clear, we had an initial calibration session where they test-rated a small portion of the data (around 10 datapoints which were then omitted from the results) and asked us any clarifying questions. For each style, we compare outputs from our method plus the three baselines for 50 sentences.

Each tuple was scored by three raters (3,600 ratings total) on the following three axes which are standard to textual style transfer (Mir et al., 2019): **(1) transfer strength** (the amount that the output actually matches the target style), **(2) semantic preservation** (whether the underlying meaning of the output text, aside from style, matches that of the input), and **(3) fluency** (whether the text is coherent and could have been written by a proficient English speaker). Following Sakaguchi and Van Durme (2018), transfer strength and semantic preservation were rated on a scale from 1–100. A screenshot of the evaluation UI is shown in Figure 5 in the Appendix. Note that the guidelines for semantic preservation are not standardized in prior literature (Briakou et al., 2021); while some evaluations are strict that the outputs cannot contain any more information than the inputs, we asked the annotators not to penalize for meaning transformations which are necessary for the specified transformation. We use *dialog-LLM*, and compare it with three other methods: **(1) zero-shot** (a baseline), **(2) paraphrase** (our normal augmented zero shot prompt, but with the target style of *"paraphrased"*, as a control) and **(3) human** (ground-truth transformations written by the authors).

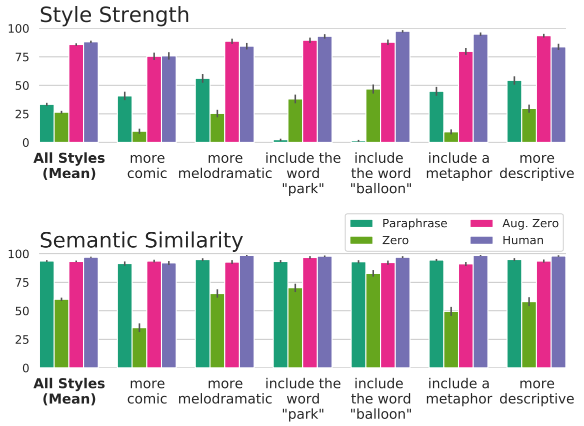


Figure 2: Human evaluation of style transfer for six atypical styles. Our method is rated comparably to the human-written ground truth. Error bars show Standard Error of the Mean. Evaluation of fluency is shown in Figure 4 in the Appendix.

Figure 2 shows these results. We found that the outputs of our method were rated almost as highly as the human-written ground truth for all three evaluations. The zero-shot baseline performed the worst in all categories: 25.4% of the time, it did not return a valid response at all (see §6), compared with 0.6% for augmented zero shot. The strong performance of the paraphrase baseline at fluency and semantic similarity shows that large LMs are capable of generating high quality text that remains true to the input sentence's meaning. Overall, the average length of the input sentences was 66 characters, whereas the average length of augmented zero-shot outputs was 107 characters. For context, human paraphrase outputs were 82 characters.

For a subset of the tasks, some automatic evaluation was also possible. We found that the "*balloon*" and "*park*" transformations successfully inserted the target word 85% of the time. For "*more descriptive*" and "*include a metaphor*" the transformed text was, as expected, longer than the original (by 252% and 146% respectively, compared with 165% and 146% for human baselines).

4.2 Standard Styles

To better contextualize the performance of our method with prior methods, we also generated outputs for two standard style transfer tasks: sentiment and formality. Figure 3 shows human evaluations (same setup as before) for our outputs as well as the outputs from two popular prior style transfer methods, Unsup MT (Prabhumoye et al., 2018) and Dual RL (Luo et al., 2019). The outputs from our method were rated comparably to both human generated responses and the two prior methods, using the same rating setup as the non-standard styles, with six outputs and baselines for four styles across 50 sentences, rated independently by three raters, totalling 3,000 total ratings.

Furthermore, following Li et al. (2018) and Sudhakar et al. (2019), we perform automatic evaluation for sentiment style transfer since there are classifiers available for these styles. We note that although automatic evaluations can diverge from human ratings, they can still be a good proxy as we could not perform human evaluation against every prior method due to time and resource constraints. We automatically evaluate **(1) transfer strength** using a sentiment classifier from HuggingFace Transformers (Wolf et al., 2020), **(2) semantic similarity** to human examples provided by Luo et al. (2019) via BLEU score, and **(3) fluency** via perplexity, as measured by GPT-2 (117M).

Table 2 shows these automatic evaluations, with four main takeaways. First, augmented zero-shot prompting achieves high accuracy and low perplexity compared with baselines. The BLEU scores, however, are low, which we believe is because it tends to add additional information to generated sentences (see Appendix B for a deeper analysis). Second, we apply augmented zero-shot learning to GPT-3 175B; these results indicate that augmented zero-shot learning generalizes to another large language model. Third, we vary model size for GPT-3 models, finding that larger size greatly improves style transfer. Fourth, for *LLM* and *LLM-dialog*, we find that augmented zero-shot learning substantially outperforms vanilla zero-shot learning and almost reaches the accuracy of five-shot learning.

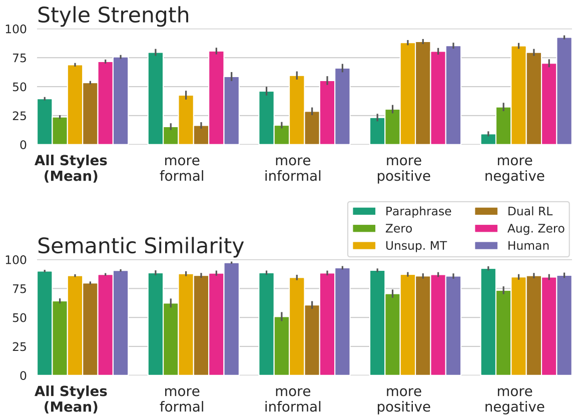


Figure 3: Human evaluation of sentiment and formality transfer. Our method is rated comparably to human-written ground truth as well as prior methods. Error bars show Standard Error of the Mean. Unsup. MT is Prabhumoye et al. (2018); Dual RL is Luo et al. (2019).

5 Potential of Arbitrary Styles

One promising application of augmented zero-shot learning is an AI-powered writing assistant that can allow writers to transform their text in arbitrary ways that the writer defines and controls. As a qualitative case study to explore what arbitrary re-write styles may be requested, we built an AI-assisted story-writing editor with a "rewrite as" feature that uses our augmented few-shot method. Our editor has a freeform text box for users to specify how they would like a selection of their story to be rewritten (see Figure 6 in the Appendix). We asked 30 people from a creative writing group to use our UI to write a 100-300 word story, collecting 333 rewrite requests in total. Table 3 shows a subset of these, which were as diverse as asking for the text "*to be about mining*" or "*to be less diabolical*."

6 Limitations and Failure Modes

This section details several qualitative limitations with our method.

Unparsable answers A frequent problem that arises when using large LMs for other NLP tasks is their outputs cannot be automatically parsed into usable answers. For example, when given a prompt like `"Here is some text: that is an ugly dress. Here is a rewrite of the text, which is more positive"` *LLM-Dialog* might return something like `"Sounds like you are a great writer!"` Similar error modes exist for *LLM*, which might output something like `"Here are more writing tips and tricks."` Other

	Acc	BLEU	PPL
SUPERVISED METHODS			
Cross-alignment (Shen et al., 2017)	73.4	17.6	812
Backtrans (Prabhumoye et al., 2018)	90.5	5.1	424
Multidecoder (Fu et al., 2018)	50.3	27.7	1,703
Delete-only (Li et al., 2018)	81.4	28.6	606
Delete-retrieve (Li et al., 2018)	86.2	31.1	948
Unpaired RL (Xu et al., 2018)	52.2	37.2	2,750
Dual RL (Luo et al., 2019)	85.9	55.1	982
Style transformer (Dai et al., 2019)	82.1	55.2	935
INFERENCE-ONLY METHODS			
GPT-3 ada, aug zero-shot	31.5	39.0	283
GPT-3 curie, aug zero-shot	53.0	48.3	207
GPT-3 da vinci, aug zero-shot	74.1	43.8	231
LLM: zero-shot	69.7	28.6	397
five-shot	83.2	19.8	240
aug zero-shot	79.6	16.1	173
LLM-dialog: zero-shot	59.1	17.6	138
five-shot	94.3	13.6	126
aug zero-shot	90.6	10.4	79

Table 2: Comparing augmented zero-shot prompting with supervised style transfer methods on the Yelp sentiment style transfer dataset using automatic evaluation. Acc: accuracy; PPL: perplexity. The inference-only table shows our method applied to 3 different sizes of GPT-3, plus our own LLM.

to be a little less angsty • to be about mining • to be better written • to be less diabolical • to be more absurd • to be more adventurous • to be more Dickensian • to be more emotional • to be more magical • to be more melodramatic • to be more philosophical • to be more revolutionary • to be more surprising • to be more suspenseful • to be more technical • to be more whimsical • to be warmer • to fit better grammatically with the rest of the story • to make more sense

Table 3: Requests in the form of "*Rewrite this...*" made by real users to a large LM-powered text editor. For the full set of unique requests, see Table 5 in the Appendix.

times, the response contains correct information, but it cannot be automatically parsed (e.g., `“a good rewrite might be to say that the dress is pretty.”`) In hindsight, these outputs make a lot of sense: most of the training data of large LMs is not well-formatted pairs of inputs and outputs (Reynolds and McDonell, 2021). See §A for how we dealt with these issues.

Hallucinations Large LMs are known to hallucinate text content; we saw this happen frequently for style transfer. While this is an advantage in some contexts like creative writing, it is undesirable for applications like summarization.

Inherent style trends We also noticed that even our *"paraphrase"* baseline, where the model was simply asked to rewrite the input sentence, was rated highly for style strength for a few styles, including *"more formal"* and *"more melodramatic"*. This implies that our method's generations generally trend toward these styles. A direction for future work would be to see what styles and qualities of text our method (and large LMs in general) are inherently more likely to produce.

Less reliable than trained methods For style transfer tasks that have available training data, prior methods that either train or finetune on that data are going to be inherently more reliable at producing text that looks like their training data. This can be observed in the lower BLEU scores our method achieves than trained methods, despite comparable transfer accuracy (Section B). Thus, augmented zero-shot learning offers less fine-grained controllability in the properties of the style-transferred text than methods which see task-specific training data.

Large LM safety concerns Large LMs themselves come with their own host of difficulties, barriers to entry, and potential safety concerns as discussed by Bender et al. (2021), which are also valid for this style transfer method. However, we also think that this method can be a useful tool in exploring and exposing the safety and boundaries of these models themselves: what happens if we try to force the large LM to make a text "more racist", "more sexist", or "more incendiary"? It is important to keep pushing these models to their boundaries to see where they fail and where problems arise, and specific use cases that show a broader range of the model's capabilities also show a broader range of its failure modes.

7 Conclusions

We introduced augmented zero-shot learning, which we find shows shows strikingly promising performance considering its simplicity. This prompting paradigm moves the needle in text style transfer by expanding the range of possible styles beyond the currently limited set of styles for which annotated data exists. More broadly, we also hope that the strategy of prompting a large LM with non-task specific examples can inspire new inference-only methods for other NLP tasks.

References

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Gwern Branwen. 2020. GPT-3 creative fiction.

Eleftheria Briakou, Sweta Agrawal, Ke Zhang, Joel R. Tetreault, and Marine Carpuat. 2021. A review of human evaluation for style transfer. *CoRR*, abs/2106.04747.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. 2019. Style transformer: Unpaired text style transfer without disentangled latent representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5997–6007, Florence, Italy. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Zhiqiang Hu, Roy Ka-Wei Lee, and Charu C. Aggarwal. 2020. Text style transfer: A review and experiment evaluation. *CoRR*, abs/2010.12742.

Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2020. Deep learning for text style transfer: A survey. *CoRR*, abs/2011.00416.

Zhijing Jin, Di Jin, Jonas Mueller, Nicholas Matthews, and Enrico Santus. 2019. IMaT: Unsupervised text attribute transfer via iterative matching and translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3097–3109, Hong Kong, China. Association for Computational Linguistics.

Kalpesh Krishna, John Wieting, and Mohit Iyyer. 2020. Reformulating unsupervised style transfer as paraphrase generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 737–762, Online. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *CoRR*, abs/1808.06226.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana. Association for Computational Linguistics.

Dayiheng Liu, Jie Fu, Yidan Zhang, Chris Pal, and Jiancheng Lv. 2020. Revision in continuous space: Unsupervised text style transfer without adversarial learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8376–8383.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Ruibo Liu, Chenyan Jia, and Soroush Vosoughi. 2021b. A transformer-based framework for neutralizing and reversing the political polarity of news articles. *Proc. ACM Hum.-Comput. Interact.*, 5(CSCW1).

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Xu Sun, and Zhifang Sui. 2019. A dual reinforcement learning framework for unsupervised text style transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5116–5122. ijcai.org.

Aman Madaan, Amrith Setlur, Tanmay Parekh, Barnabas Poczos, Graham Neubig, Yiming Yang, Ruslan Salakhutdinov, Alan W Black, and Shrimai Prabhumoye. 2020. Politeness transfer: A tag and generate approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1869–1881, Online. Association for Computational Linguistics.

Remi Mir, Bjarke Felbo, Nick Obradovich, and Iyad Rahwan. 2019. Evaluating style transfer for text. *CoRR*, abs/1904.02295.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876, Melbourne, Australia. Association for Computational Linguistics.

Raul Puri and Bryan Catanzaro. 2019. Zero-shot text classification with generative language models. *arXiv preprint arXiv:1912.10165*.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 129–140, New Orleans, Louisiana. Association for Computational Linguistics.

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm.

Parker Riley, Noah Constant, Mandy Guo, Girish Kumar, David C. Uthus, and Zarana Parekh. 2021. Textsettr: Label-free text style extraction and tunable targeted restyling. *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL)*.

Keisuke Sakaguchi and Benjamin Van Durme. 2018. Efficient online scalar annotation with bounded support. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 208–218, Melbourne, Australia. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. 2019. "Transforming" delete, retrieve, generate approach for controlled text style transfer. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3269–3279, Hong Kong, China. Association for Computational Linguistics.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Orion Weller, Nicholas Lourie, Matt Gardner, and Matthew E. Peters. 2020. Learning from task descriptions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1361–1375, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jingjing Xu, Xu Sun, Qi Zeng, Xiaodong Zhang, Xuancheng Ren, Houfeng Wang, and Wenjie Li. 2018. Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 979–988, Melbourne, Australia. Association for Computational Linguistics.

Peng Xu, Yanshuai Cao, and Jackie Chi Kit Cheung. 2020. On variational learning of controllable representations for text without supervision. *Proceedings of the International Conference on Machine Learning (ICML)*, abs/1905.11975.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Proceedings of the Conference on Neural Information Processing Systems*.

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

Appendix

A Prompt Selection

A promising new area of prompt engineering has arisen to address the failure modes discussed above, specifically the invalid or unparseable answers. Reynolds and McDonell (2021) find that prompting a model for a task is more akin to locating an already-learned task than truly learning a new one. Moreover, they emphasize that prompt engineering is mostly about avoiding various failure cases such as those described above. In this work, we use delimiters ("{" and "}") to help avoid these types of errors, giving scores of zero when there was no valid responses with such delimiters. There are other delimiters that could be used (e.g., quotes, "(" and ")", "<" and ">", newlines with a colon (as used by GPT-3), etc. We chose curly braces as they were 1) likely to occur in the training data as delimiters in other contexts and 2) not frequently part of the input sentence itself. We also use a second person prompt template for the dialog, which yielded better results as it was more similar to the training data. Exploring these options more quantitatively would be an interesting direction for future work. Because the performance of prompting can vary depending on the exact language of the prompt (Reynolds and McDonell, 2021), we compare four variations of prompts for sentiment: "*more positive/negative*," "*happier/sadder*," "*more optimistic/pessimistic*," and "*more cheerful/miserable*." As shown in Table 4 in the Appendix, performance differed across the four prompts, but we found them comparable.

Model / prompt wording	Acc	Bleu	PPL
LLM			
"more positive/negative"	76.3	14.8	180
"happier/sadder"	62.6	15.5	173
"more optimistic/pessimistic"	69.7	14.1	143
"more cheerful/miserable"	74.5	15.7	186
LLM-Dialog			
"more positive/negative"	90.5	10.4	79
"happier/sadder"	85.9	9.6	90
"more optimistic/pessimistic"	85.8	10.2	79
"more cheerful/miserable"	88.8	11.4	93

Table 4: Comparing variations of augmented zero-shot learning prompt wording for sentiment style transfer.

into paragraphs • to be a bit clearer • to be a little less angsty • to be a word for a song • to be about mining • to be about vegetables • to be better written • to be less descriptive • to be less diabolical • to be more absurd • to be more adventurous • to be more angry • to be more cheerful • to be more descriptive • to be more Dickensian • to be more emotional • to be more fancy • to be more flowery • to be more interesting • to be more joyful • to be more magical • to be more melodramatic • to be more philosophical • to be more revolutionary • to be more scary • to be more subtle • to be more surprising • to be more suspenseful • to be more technical • to be more violent • to be more whimsical • to be warmer • to fit better grammatically with the rest of the story • to make more sense • to use a more interesting word • with a few words

Table 5: Full results for requests in the form of "*Rewrite this...*" made by users to a large LM-powered text editor.

B Low BLEU for *LLM* Outputs

As we saw in Table 2, the outputs of our model had low BLEU scores with respect to human generated outputs, while simultaneously having high semantic similarity in human evaluations. Based on qualitative examination of outputs, we believe that this is because model outputs often, despite having high semantic similarity with the source sentence, used different language from human annotations. For instance, for transferring the sentiment of "*ever since joes has changed hands it's just gotten worse and worse*" to positive sentiment, our augmented zero-shot learning model outputted "the establishment has continued to provide excellent service, improving steadily since its change of ownership." This will have low BLEU with the ground truth with respect to human references, which is simply "*ever since joes has changed hands it's just gotten better and better*."

Though we do not see this as an inherent problem, increasing the BLEU for the purposes of comparison can be done in an easy way via candidate selection, as our model returns sixteen possible continuations. In applications for which we prefer model outputs to have high lexical similarity to the source sentence, we could select the candidate of the sixteen with the highest BLEU score compared with the original source sentence. We find that this candidate selection step can substantially improve the BLEU score with the ground truth target sentences, as we show in Table 8.

C Further Related Work

Style transfer has gained increasing attention in the NLP landscape, for which neural models have been trained to perform style transfer for styles including sentiment, formality, politeness, gender, and politi-

Input	Model suggestion
You honor me," said Bruno. As he spoke, he began to laugh. His laugh was merry sounding and melodic. He took a deep breath. When he spoke again, his voice was low and strong. "I take this great task on and with my brothers shall lead us on to victory! **Rewrite it to be about mining.**	You honor me, said Bruno. As he spoke, he began to laugh. His laugh was deep and gravely, as if from the bowels of the earth. He took a deep breath. When he spoke again, his voice was deep and dark.'I take this great task on and with my brothers shall delve into the mines!"
I reckon,' smiled Toe, 'you'd have to say the same for icicles, wouldn't you? **Rewrite it to be more interesting.**	'I reckon,' smiled Toe, 'you'd have to say the same for icicles, wouldn't you?' These words were barely out of his mouth before the others realized exactly what they meant.
with its southern perfume **Rewrite it to be more descriptive**	A mixture of sweet honeysuckles and jasmine with a hint of tobacco and warm mesquite smoke. It was his southern perfume, with a hint of a memory of simpler times.
But Sally remained unmoved–and lonely. **Rewrite it to be more emotional**	Her tears rolled and rolled down her cheek, mixing with the saltwater around her, until the water was a rich mixture of the two. She wanted her poem to be cheerful like it used to be, but couldn't find the right words.

Table 6: Examples of users' arbitrary style transfer requests for which the model suggestion was accepted.

cal slant (Prabhumoye et al., 2018; Madaan et al., 2020; Liu et al., 2021b). We will briefly summarize the primary approaches to style transfer here, and refer the involved reader to either (Jin et al., 2020) or (Hu et al., 2020) for a survey.

Most text style transfer approaches fall in two categories. Early approaches tend to require *parallel* text data (Zhu et al., 2010; Rao and Tetreault, 2018), where every input in the source style has a corresponding output in the target style. Though this formulation elegantly fits the standard encoder–decoder paradigm, the availability of a parallel text corpus is a stringent requirement. Hence, recent text style transfer approaches have instead used *non-parallel* monostyle data (no one-to-one-mapping between instances in the source and target styles). Such methods include latent representation manipulation (Liu et al., 2020), prototype-based text editing (Li et al., 2018), and pseudo-parallel corpus construction (Jin et al., 2019). However, even non-parallel monostyle data can be hard to collect for arbitrary styles. As such, surveys have called for more research on approaches that expand the scope of supported styles and reduce the training data requirements for style transfer systems (Jin et al., 2020; Hu et al., 2020).

Several new methods tackle the challenging problem of *label-free* style transfer, which does not require a full corpus of labeled data, but rather just a few exemplars that define a style. Xu et al. (2020) use variational autoencoders for unsupervised learning of controllable representations for text. Riley et al. (2021) extract a style vector from a set of target texts and use this vector to condition the decoder to perform style transfer to a target style. These approaches have a similar goal to ours in terms of expanding the scope of possible style transfers. However, they are different in two main ways. First, they require a fully specialized model, where our method can be applied out-of-the-box with something like GPT-3. This can either be a strength or weakness, depending on the availability of such a model. Second, they require exemplars to define a style rather than a plain text description.

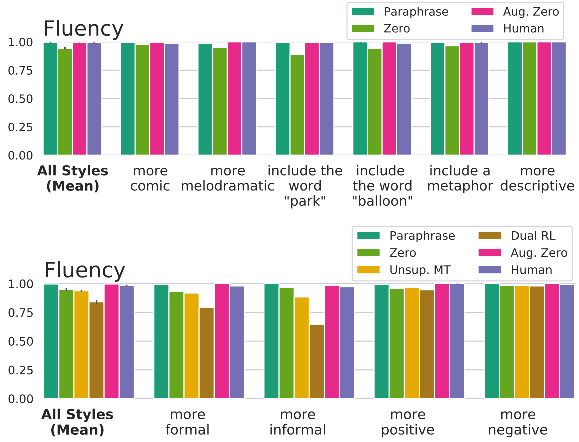


Figure 4: Human evaluation of fluency for style transfer for six atypical styles. Error bars show standard error of the mean.

Augmented Zero-shot Prompt: LLM

```
Here is some text:  {When the doctor asked Linda to take the medicine, he smiled and gave her a lollipop.}.  Here
is a rewrite of the text, which is more scary.  {When the doctor told Linda to take the medicine, there had been
a malicious gleam in her eye that Linda didn't like at all.} Here is some text:  {they asked loudly, over the
sound of the train.}.  Here is a rewrite of the text, which is more intense.  {they yelled aggressively, over the
clanging of the train.} Here is some text:  {When Mohammed left the theatre, it was already dark out}.  Here is
a rewrite of the text, which is more about the movie itself.  {The movie was longer than Mohammed had expected,
and despite the excellent ratings he was a bit disappointed when he left the theatre.} Here is some text:  {next
to the path}.  Here is a rewrite of the text, which is about France.  {next to la Siene} Here is some text:  {The
man stood outside the grocery store, ringing the bell.}.  Here is a rewrite of the text, which is about clowns.
{The man stood outside the circus, holding a bunch of balloons.} Here is some text:  {the bell ringing}.  Here is
a rewrite of the text, which is more flowery.  {the peales of the jangling bell} Here is some text:  {against the
tree}.  Here is a rewrite of the text, which is include the word "snow".  {against the snow-covered bark of the
tree} Here is some text:  {That is an ugly dress}.  Here is a rewrite of the text, which is more positive."
```

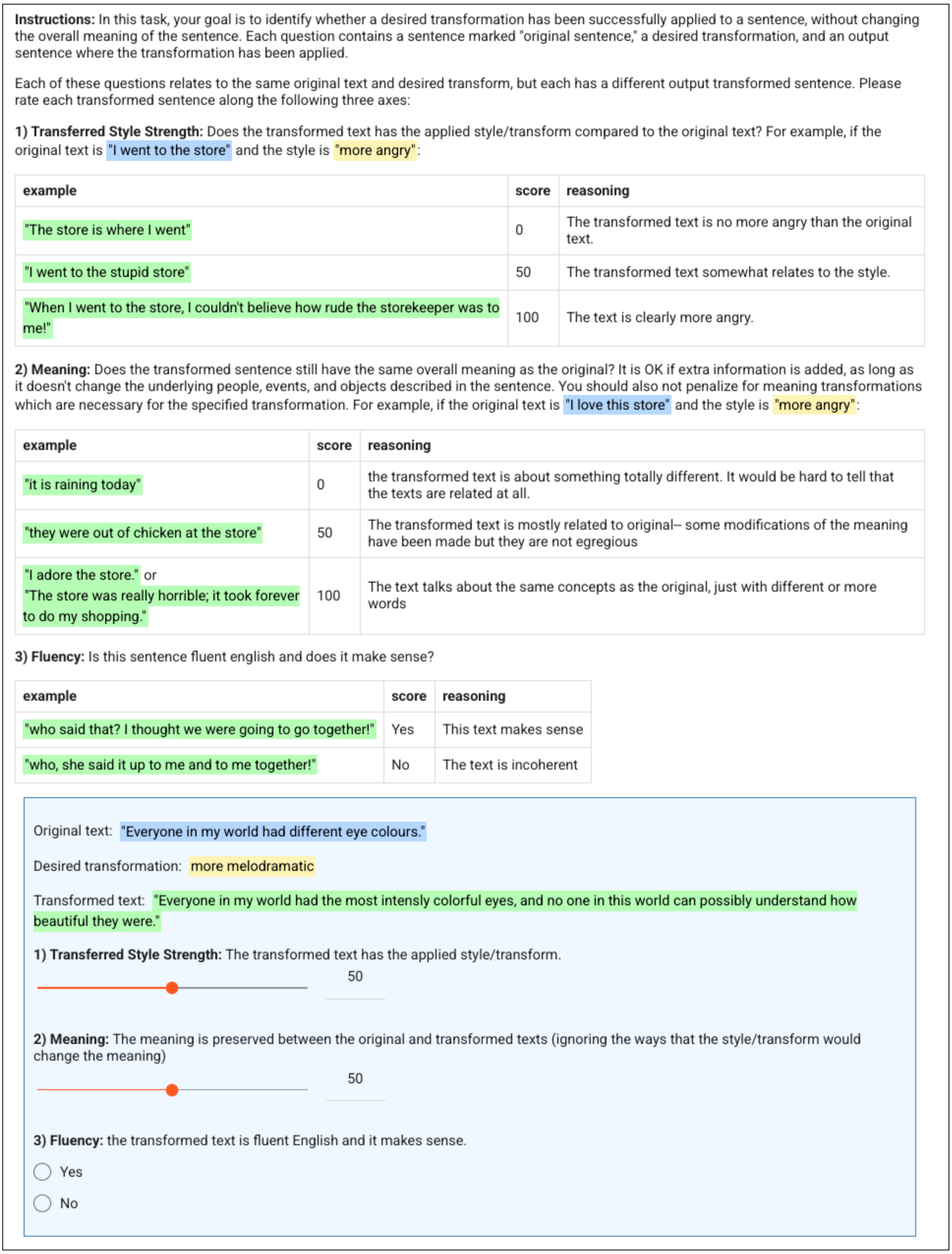
Augmented Zero-shot Prompt: LLM-dialog

```
> Here is some text:  {When the doctor asked Linda to take the medicine, he smiled and gave her a lollipop.}.
Rewrite it to be more scary.
> {When the doctor told Linda to take the medicine, there had been a malicious gleam in her eye that Linda didn't
like at all.}
> Here is some text:  {they asked loudly, over the sound of the train.}.  Rewrite it to be more intense.
> {they yelled aggressively, over the clanging of the train.}
> Here is some text:  {When Mohammed left the theatre, it was already dark out}.  Rewrite it to be more about the
movie itself.
> {The movie was longer than Mohammed had expected, and despite the excellent ratings he was a bit disappointed
when he left the theatre.}
> Here is some text:  {next to the path}.  Rewrite it to be about France.
> {next to la Siene}
> Here is some text:  {The man stood outside the grocery store, ringing the bell.}.  Rewrite it to be about
clowns.
> {The man stood outside the circus, holding a bunch of balloons.}
> Here is some text:  {the bell ringing}.  Rewrite it to be more flowery.
> {the peals of the jangling bell}
> Here is some text:  {against the tree}.  Rewrite it to be include the word "snow".
> {against the snow-covered bark of the tree}
> Here is some text:  {That is an ugly dress}.  Rewrite it to be more positive."
```

Table 7: In black, we show the exact augmented-zero shot prompts used in our experiments, for *LLM* and GPT-3 (top), and for *LLM-Dialog* (bottom). As shown, for *LLM-Dialog*, we replaced “`Here is a rewrite of the text, which is`” with “`Rewrite it to be`”. Each line starting with “>"" above was passed in as an individual dialog turn. The blue shows how an input text and goal style are concatenated to the few-shot prompt in order to produce final model output. Note that we can achieve high accuracy even though the prompt formulation resulted in some minor grammatical errors for some styles (e.g., “`rewrite it to be include the word 'snow'`”). Text versions of these prompts can be downloaded at `https://bit.ly/3fLDuci`.

	Acc	BLEU	PPL
LLM-128B			
Zero-shot	69.7	28.6	397
+ cand. select.	31.4	61.5	354
Five-shot	83.2	19.8	240
+ cand. select.	61.5	55.6	306
Augmented zero-shot	79.6	16.1	173
+ cand. select.	65.0	49.3	292
LLM-128B-dialog			
Zero-shot	59.1	17.6	138
+ cand. select.	46.8	24.2	166
Five-shot	94.3	13.6	126
+ cand. select.	81.3	47.6	345
Augmented zero-shot	90.6	10.4	79
+ cand. select.	73.7	40.6	184

Table 8: Sentiment style transfer results with candidate selection (cand. select.). Candidate selection means that of the sixteen examples returned by our model, we choose the one with the highest BLEU with the source sentence.



Instructions: In this task, your goal is to identify whether a desired transformation has been successfully applied to a sentence, without changing the overall meaning of the sentence. Each question contains a sentence marked "original sentence," a desired transformation, and an output sentence where the transformation has been applied.

Each of these questions relates to the same original text and desired transform, but each has a different output transformed sentence. Please rate each transformed sentence along the following three axes:

1) Transferred Style Strength: Does the transformed text has the applied style/transform compared to the original text? For example, if the original text is "I went to the store" and the style is "more angry":

example	score	reasoning
"The store is where I went"	0	The transformed text is no more angry than the original text.
"I went to the stupid store"	50	The transformed text somewhat relates to the style.
"When I went to the store, I couldn't believe how rude the storekeeper was to me!"	100	The text is clearly more angry.

2) Meaning: Does the transformed sentence still have the same overall meaning as the original? It is OK if extra information is added, as long as it doesn't change the underlying people, events, and objects described in the sentence. You should also not penalize for meaning transformations which are necessary for the specified transformation. For example, if the original text is "I love this store" and the style is "more angry":

example	score	reasoning
"it is raining today"	0	the transformed text is about something totally different. It would be hard to tell that the texts are related at all.
"they were out of chicken at the store"	50	The transformed text is mostly related to original-- some modifications of the meaning have been made but they are not egregious
"I adore the store." or "The store was really horrible; it took forever to do my shopping."	100	The text talks about the same concepts as the original, just with different or more words

3) Fluency: Is this sentence fluent english and does it make sense?

example	score	reasoning
"who said that? I thought we were going to go together!"	Yes	This text makes sense
"who, she said it up to me and to me together!"	No	The text is incoherent

Original text: "Everyone in my world had different eye colours."

Desired transformation: more melodramatic

Transformed text: "Everyone in my world had the most intensly colorful eyes, and no one in this world can possibly understand how beautiful they were."

1) Transferred Style Strength: The transformed text has the applied style/transform.

50

2) Meaning: The meaning is preserved between the original and transformed texts (ignoring the ways that the style/transform would change the meaning)

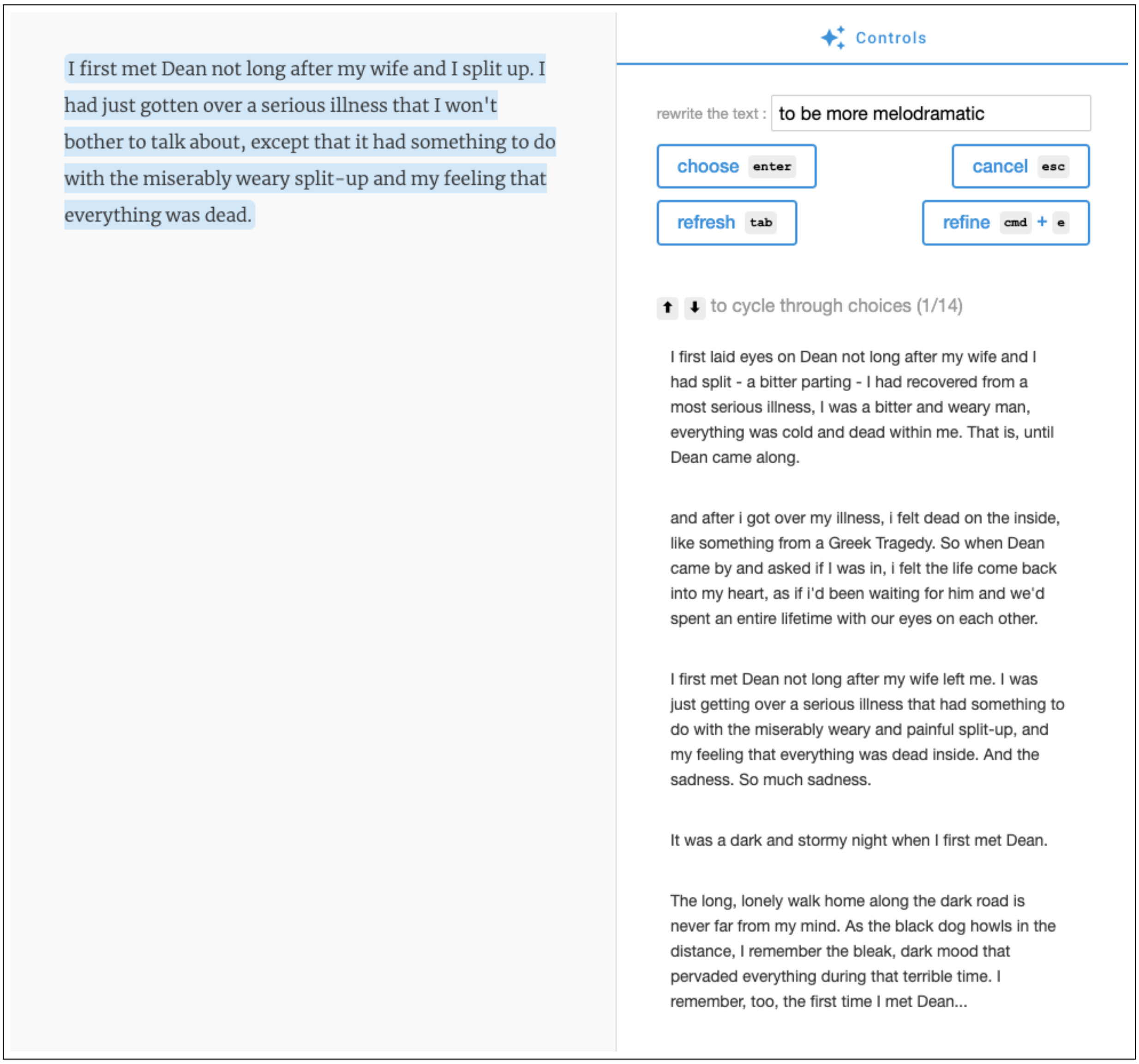
50

3) Fluency: the transformed text is fluent English and it makes sense.

○ Yes

○ No

Figure 5: The rating UI used for human evaluation. The user may be shown a number of blue squares at once with the same original text and different outputs.



I first met Dean not long after my wife and I split up. I had just gotten over a serious illness that I won't bother to talk about, except that it had something to do with the miserably weary split-up and my feeling that everything was dead.

rewrite the text : to be more melodramatic

↑ ↓ to cycle through choices (1/14)

I first laid eyes on Dean not long after my wife and I had split - a bitter parting - I had recovered from a most serious illness, I was a bitter and weary man, everything was cold and dead within me. That is, until Dean came along.

and after i got over my illness, i felt dead on the inside, like something from a Greek Tragedy. So when Dean came by and asked if I was in, i felt the life come back into my heart, as if i'd been waiting for him and we'd spent an entire lifetime with our eyes on each other.

I first met Dean not long after my wife left me. I was just getting over a serious illness that had something to do with the miserably weary and painful split-up, and my feeling that everything was dead inside. And the sadness. So much sadness.

It was a dark and stormy night when I first met Dean.

The long, lonely walk home along the dark road is never far from my mind. As the black dog howls in the distance, I remember the bleak, dark mood that pervaded everything during that terrible time. I remember, too, the first time I met Dean...

Figure 6: Screenshot AI-assisted editor with ‘Rewrite as’ feature.

Style	Inputs	Aug. Zero	Zero	Human	Paraphrase
more comic	75	116	63	97	87
more melodromatic	75	124	88	116	87
include the word “park”	75	124	72	94	87
include the word “balloon”	75	135	86	98	87
include a metaphor	75	110	74	110	87
more descriptive	75	190	105	124	87
Overall	75	133	81	107	87

Table 9: The mean length in characters of the inputs and outputs for our six atypical styles.

DiS-ReX: A Multilingual Dataset for Distantly Supervised Relation Extraction

Abhyuday Bhartiya*
Indian Institute of Technology
New Delhi, India
bhartiyabhyuday@gmail.com

Kartikeya Badola*
Indian Institute of Technology
New Delhi, India
kartikeya.badola@gmail.com

Mausam
Indian Institute of Technology
New Delhi, India
mausam@cse.iitd.ac.in

Abstract

Our goal is to study the novel task of distant supervision for *multilingual* relation extraction (Multi DS-RE). Research in Multi DS-RE has remained limited due to the absence of a reliable benchmarking dataset. The only available dataset for this task, RELX-Distant (Köksal and Özgür, 2020), displays several unrealistic characteristics, leading to a systematic overestimation of model performance. To alleviate these concerns, we release a new benchmark dataset for the task, named DiS-ReX. We also modify the widely-used bag attention models using an mBERT encoder and provide the first baseline results on the proposed task. We show that DiS-ReX serves as a more challenging dataset than RELX-Distant, leaving ample room for future research in this domain.

1 Introduction

Relation Extraction (RE) identifies the relation r between a pair of entities (e_1, e_2) given some text mentioning both of them. To avoid large manual annotation, RE is often trained via distant supervision (DS-RE) (Mintz et al., 2009). DS-RE uses facts $r(e_1, e_2)$ in an existing KB to associate a label r with the bag containing all sentences that mention e_1 and e_2. Research in DS-RE has been mostly monolingual and limited to English. Our goal is to study multilingual RE via distant supervision (Multi DS-RE). We expect multilingual RE models to have several benefits over monolingual RE. First, training data from multiple languages may be pooled to create a large dataset, enabling cross-lingual knowledge transfer (Zoph et al., 2016; Feng et al., 2020). Second, it may encourage RE models to be consistent across languages (Lin et al., 2017), e.g., extraction of a fact already seen in one language should be easier in another.

To the best of our knowledge, RELX-Distant (Köksal and Özgür, 2020) is currently the only dataset available for Multi DS-RE, but even so, it has never been evaluated as a benchmark for the task. Our analysis reveals that it suffers from a poor selection of relation classes. Firstly, there are no examples of NA class (sentences with no relation between the two entities). Therefore, a model trained on RELX-Distant would find limited utility in any real world setting. Secondly, its choice of relation classes is highly disjoint, resulting in an absence of instances with multiple labels (unusual for a DS-RE dataset). Finally, it is highly imbalanced – even though it has 24 relation classes, over 50% bags belong to just one "country" relation.

Owing to these attributes, we observe that models trained on RELX-Distant end up classifying the instances of the minority class based on just the entity type information. Due to high skew, such mistakes have negligible impact on evaluation scores and the model achieves an AUC of 0.99 after only 5 training epochs. Such numbers are unheard of, especially when compared to benchmarking datasets in mono-lingual RE (mono-lingual variant of the same architecture obtains an AUC of 0.83 when trained and tested on the GDS dataset (Jat et al., 2018).

In response, we contribute a more realistic benchmark dataset for the task called DiS-ReX. Our dataset has over 1.8 million sentences in four languages: English, French, Spanish and German. It has 37 relation types including 1 No-Relation (NA) class and also has instances with multiple labels similar to the widely-used New York Times (NYT) dataset for English DS-RE (Riedel et al., 2010), thus comparing favorably to RELX-Distant.

We also adopt state-of-the-art DS-RE models in the multilingual setting by using the mBERT encoder (Devlin et al., 2019), to create a strong baseline for this task.

We achieve an AUC of 0.82 and a Micro-F1 of 0.76, suggesting that the dataset is not trivial to optimize on, and could act as a good benchmark

* Equal Contribution

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 849 - 863
May 22-27, 2022 ©2022 Association for Computational Linguistics

Language	#sentences	# bags	# non-NA bags	Average non-NA bag-size
English	532499	216806	66932	4.50
French	409087	226418	83951	2.88
Spanish	456418	229512	80706	2.88
German	438315	194942	45908	3.48

Table 1: Key statistics for DiS-ReX

for the task. We publicly release DiS-ReX and the baseline.[1]

2 Related Work

Supervised RE datasets such as ACE05 (Walker et al., 2006) and KLUE (Park et al., 2021) are generally small, owing to the supervision needs per relation. Distant supervision (Mintz et al., 2009) is a popular alternative to large-scale human annotation, but necessitate more complex models to handle dataset noise. The standard English DS-RE dataset is New York Times (NYT) corpus (Riedel et al., 2010), which has served as the benchmark for research over the years. DS-RE models have evolved to use multi-instance learning (Hoffmann et al., 2011), multi-label learning (Surdeanu et al., 2012), corrections for false negatives (Ritter et al., 2013), and neural models such as piecewise CNNs (Zeng et al., 2015), intra-bag attention (Lin et al., 2016), and reinforcement learning (Qin et al., 2018).

Lin et al. (2017) and Wang et al. (2018) propose extensions of bag-attention models for bilingual (English-Mandarin) datasets. However, their adoption to multiple languages has been lacking, due to absence of a reliable multilingual dataset. Although RELX-Distant is the only Multilingual DS-RE dataset so far, it wasn't originally used for Multi DS-RE task but to pre-train a model that gets fine-tuned for *supervised* RE task.

Contemporary to our work, other multilingual RE datasets and methods are being developed. These include a dataset for joint entity and relation extraction (Seganti et al., 2021), a model for multilingual KB completion (Singh et al., 2021), and an approach for automatic construction of cross-lingual training data for Open IE (Kolluru et al., 2022). Our proposed dataset, DiS-ReX, has already been used for further research on the Multilingual DS-RE task (Rathore et al., 2022).

3 Dataset Curation

All distant supervision datasets are curated by aligning known KB facts with sentences in a large corpus. We follow the same for DiS-ReX, while paying attention to cross-lingual normalization, and overall data and language statistics.

First, we harvest a large number of sentences from English, French, Spanish and German Wikipedias.[2] We use DBPedia language editions (Lehmann et al., 2015) for our KB – this gives us good coverage of entities that are local to different language speakers. DBPedia entities are associated with Wikidata IDs, which are normalized across languages. This enables us to fuse these DBPedia KBs and establish equivalence between entities like *USA* and *Estados Unidos de América*.

Next, we use a language-specific NER tagger, (we use the *md* variant of spaCy (Honnibal et al., 2020) NER taggers for each language), returning a rich set of sentences. In contrast, RELX-Distant finds entity mentions using Wikipedia hyperlinks. This severely limits its pool of sentences, since often only the first mention of an entity in a Wiki document has a hyperlink while others do not.

Linking each mention with its entity can be challenging, due to unavailability of high-quality entity linking software for every language. We take the pragmatic approach of using simple string matching, but only on the subset of entities that have an unambiguous surface form (or alias) in our fused KB. This maintains scalability to many languages, while ensuring high enough precision of linking.

For each entity-pair, we create a language-specific bag of all sentences that mention both. We also search for all relations between them in our fused KB. We associate the bag with all those relation labels, or "NA", if no relation is found.

Our next steps select a balanced subset of this dataset, so that it can serve as a good benchmark for Multi DS-RE. We first select the subset of relations that have at least 50 bags in all languages.

[1] https://github.com/dair-iitd/DiS-ReX

[2] Our pipeline applies to non-Wikipedia sentences too.

This yields the 36 positive relation types used in our data. For each relation type, we limit the number of bags in a language to a max of 10,000. This helps curb the skew due to highly frequent relations such as $country$ and $birthPlace$. During this filtering, we ensure that bags of entity pairs common across more than one language are not removed, so that we have an abundant number of cross-lingual bags. Models can take advantage of such bags for establishing representation consistency across languages (Wang et al., 2018). Finally, we add bags of entity pairs that have no relation between them. Similar to NYT dataset, "NA" is the majority class in DiS-ReX (kept at roughly 70%).

Hence, we obtain a dataset with over 1.8 million sentences, and over 250,000 (non-NA) bags (see table 1 for more statistics). The 36 relations include frequent relations between persons, locations and organizations (e.g., *capital, headquarter, works-at*), and also some relations with fine-grained types such as *bandMember, starring* and *recordLabel*.

We estimate the percentage of bags satisfying "at-least one" assumption by manually labelling sentences across 50 randomly selected bags. We find that 82% of the bags satisfy "at-least one" assumption. For the test set of NYT Corpus, this percentage is close to 62% (Zhu et al., 2020)

Finally, we create train-dev-test splits by splitting the bags in the ratio 70 : 10 : 20. While splitting we ensure that entity-pairs in three sets are mutually exclusive, so the model does not extract by memorizing a fact.

4 Experiments and Data Analysis

4.1 Comparison: DiS-ReX vs. RELX-Distant

We now compare the two datasets: DiS-ReX and RELX-Distant. We find that the our dataset showcases several desirable properties expected from a challenging DS-RE dataset, including the presence of NA relations, inverse relations, multi-label bags, and better class balance.

70% of bags in DiS-ReX are NA bags, whereas RELX-Distant has none. We also note that a few relation pairs (from our 36 relations) represent inverses of each other, e.g., {*influenced by, influenced*}, {*successor, predecessor*}, and {*associatedBand, bandMember*}. Inverse relations test an extractor's ability to output related relations from the same bag, but with different entity ordering. RELX-Distant has no inverse relations in its relation vocabulary.

	RELX-Distant	**DiS-ReX**
Efficiency (η)	0.522	0.856
M-F1 (top 3)	94.29	82.06
M-F1 (bottom 3)	49.47	63.28

Table 2: Key statistics representing class imbalance between RELX-Distant and DiS-ReX

Lang.	**RELX-Distant**			**DiS-ReX**		
	AUC	**μF1**	**M-F1**	**AUC**	**μF1**	**M-F1**
English	0.99	0.95	0.78	0.78	0.71	0.69
French	0.99	0.96	0.79	0.81	0.75	0.68
Spanish	0.98	0.94	0.77	0.80	0.73	0.66
German	0.99	0.95	0.80	0.76	0.72	0.59
All	0.99	0.95	0.79	0.81	0.74	0.68

Table 3: Language-wise performance of mBERT + Att. μF1 and M-F1 refer to micro and macro F1 scores.

Model	**AUC**	**Micro-F1**	**Macro-F1**
PCNN+ Att	0.678	0.634	0.437
mBERT+ Att	0.806	0.741	0.676
mBERT+ MNRE	0.817	0.759	0.706

Table 4: Performance of DS-RE models on DiS-ReX

A key characteristic of DS-RE problems is that they need multi-label modeling (Surdeanu et al., 2012), since multiple relations commonly exist between an entity pair. RELX-Distant has no such bags, primarily because its choice of relation types is such that almost no entity-pair can have multiple relations. E.g., its Person-Person relations are *mother, spouse, father, sibling, partner*, where multi-label bags are highly unlikely. In contrast, DiS-ReX has 21,642 bags that have more than one relation label. As an example, the entity pair (*Isaac Newton, England*) is associated with four relations – *birthPlace, country, deathPlace* and *nationality*.

To compare the imbalance amongst non-NA relation classes in DiS-ReX and RELX-Distant, we calculate normalized entropy (Shannon, 1948), also known an Efficiency (η). Value closer to 1 indicates that the class-wise distribution is closer to the uniform distribution. Results in Table 2 indicate that DiS-ReX is a more balanced dataset (more details regarding calculation of η in appendix)

4.2 Baseline Performance

We implement three DS-RE baselines for our DiS-ReX dataset. Our first baseline is PCNN+Att (Lin et al., 2016), which uses a piece-wise CNN as the sentence encoder and performs bag-level multi-label classification using Intra-Bag attention. In this model, each language is trained and tested upon separately. Inspired by Ni and Florian (2019), we extend this to design a second baseline, mBERT+Att. It replaces PCNN encoders with a shared mBERT encoder (Devlin et al., 2019) and retains the intra-bag attention architecture for constructing the bag representation. Our last baseline is mBERT+MNRE, which adapts the MNRE model (Lin et al., 2017) to our setting. MNRE introduced cross-lingual attention for bilingual RE. We extend this attention module to more than two languages and also replace its language-specific CNN encoders with a shared mBERT encoder. More details on baselines and training are in appendix.

We first compare mBERT+Att model on both DiS-ReX and RELX-Distant in Table 3. We find that RELX-Distant achieves an unreasonably high AUC and micro-F1. Since Micro-F1 may be overwhelmed by a few highly frequent relations, we also report Macro-F1 scores. Even the Macro-F1 scores of RELX-Distant are over 10 pt higher, suggesting that DiS-ReX is a more challenging dataset for our task. We also report the Macro-avg of F1 scores of 3 most frequent and 3 least frequent classes of both the datasets in Table 2. The performance drops by 45pts in RELX-Distant, more than double the decrease observed in our dataset, corroborating that the RELX-Distant model is not learning infrequent relations effectively. For that model, we notice that the person-person relation types, which are minority classes, obtain the lowest F1 scores. It gets confused between *mother* and *spouse* or between *father* and *sibling*. In some cases, the confidence is as high as 95% on such errors. This suggests that the model is making predictions based solely on head-tail entity types in instances belonging to the person-person relation classes. But, such mistakes depress the Micro-F1 and AUC scores only negligibly, due to severe class imbalance. Thus, the high scores do not reflect high model quality.

We report results of three models on DiS-ReX in Table 4 – mBERT+MNRE achieves 0.82 AUC and 0.76 micro-F1, establishing the best baseline performance on our task.

4.3 Error Analysis

We find that due to incorporation of NA class, multi-label bags and fine-grained relation classes, DiS-ReX offers several new challenges. We observe that on multi-label bags, micro-F1 falls drastically from roughly 0.84 (bags with 1 label) to 0.35 (4 labels), primarily due to reducing recall (statistics in Table 5).

#relations	Micro-F1	Precision	Recall
1	0.842	0.865	0.820
2	0.673	0.934	0.525
3	0.518	0.959	0.354
4	0.348	0.937	0.214

Table 5: Comparing performance of mBERT+MNRE on entity pairs with different number of labels in the ground truth in the DiS-ReX dataset

We also perform manual error analysis of 100 random and 100 most confident mistakes made by the model trained on DiS-ReX. For errors where a non-NA relation is incorrectly predicted as another, we find one major error class – highly confident mistakes in predicting closely related relation types that have high overlaps, such as {*author*, *director*}, and {*homeTown*, *birthPlace*}. Some model errors correspond to confusion in predicting inverse relations such as {*successor*,*predecessor*} and {*influenced*,*influencedBy*}. Such cases are absent in the RELX-Distant test set. We found less than 10% errors within the confident errors are due to entity disambiguation mistakes in ground truth, however, we found no such data error in the 100 random errors, suggesting that this failure mode is not the most frequent, and the test data is relatively clean.

We additionally divide the errors made on the entire test set by the best performing model into three variants.

- Type-1 Error : Model predicts a positive (Non-NA) relation label *R1* and ground label is also a positive (non-NA) relation label *R2* but *R2* is not the same as *R1*.

- Type-2 Error : Model predicts NA relation label but ground label is a positive (non-NA) relation label.

- Type-3 Error : Model predicts positive (non-NA) relation label but ground label is NA relation label.

We present the distribution of these three errors in Table 6. Predicting non-NA as NA and NA as non-NA relation make up most (55-85%) of the errors. We believe that eliminating such kinds of errors would be an important focus area in DS-RE research, especially for datasets which are better representative of real world settings.

Language	Type-1 Error (%)	Type-2 Error (%)	Type-3 Error (%)
English	44.49	31.17	24.33
French	29.69	36.14	34.15
Spanish	35.08	36.37	28.54
German	14.94	45.28	39.77

Table 6: Types of Errors made in different languages for mBERT+MNRE on DiS-ReX

4.4 Is mBERT+Att Language Agnostic?

It is believed that sharing mBERT encoder across languages is advantageous for cross-lingual transfer (Wu and Dredze, 2019). This is reflected in our experiments too where mBERT+Att strongly outperforms PCNN+Att.

mBERT+Att produces a *single* embedding for a multilingual bag, summarizing mBERT embeddings of individual sentences. We posit that for this model to achieve its true potential on DiS-ReX, mBERT encoder must learn to map all sentences to a language-agnostic representation space, or else the downstream bag attention model may get confused between intra-language and inter-language variability. We investigate this further by raising the question: is the mBERT encoder learning language agnostic embeddings?

For this we encode all sentences in multilingual bags (that contain all languages) using the encoder of trained mBERT+Att model and plot the sentence embeddings using tSNE. We show an illustrative figure for the bag (Swiss, Switzerland) in Figure 1. We find that mBERT clusters sentences of one language together, irrespective of their content (more figures in Appendix). This suggests that mBERT embeddings strongly retain language information, and are not language-agnostic.

This may prove to be a significant obstacle towards progress on our task, since the noise-filtering intra-bag attention may end up capturing variance across languages more than variance in semantics. This may also explain why mBERT+MNRE performs better, since it generates embeddings of sub-bags of each language separately, instead of a single embedding for a multilingual bag.

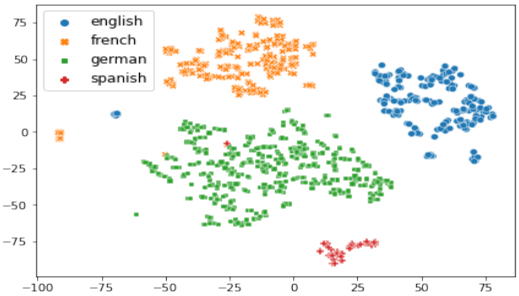


Figure 1: tSNE plot of bag (Swiss, Switzerland)

5 Conclusion

We propose DiS-ReX, a novel dataset for Multi DS-RE in 4 languages. We show that it is a more realistic and challenging benchmark compared to the existing dataset. DiS-ReX has a fairly well-represented distribution of relation types, includes instances with no-relation between entity-pairs and the relation-types selected show several real-world characteristics like inverse relations, different relations with high overlap, etc. We also publish first baseline numbers on the task of Multi DS-RE by extending existing state-of-the-art models. A detailed analysis of model performance suggests several research challenges for future: (1) learning language-agnostic sentence embeddings, (2) robustness to related relations (inverse; overlapping but semantically different), and (3) handling multi-label entity-pairs. Recently, Rathore et al. (2022) develop a multilingual DS-RE model named PARE, which reports improved performance on the DiS-ReX dataset.

Acknowledgements

This work is primarily supported by a grant from Huawei. It is also supported by grants from Google, Jai Gupta Chair professorship and a Visvesvaraya faculty award by the Govt. of India to Mausam. We thank IIT Delhi HPC facility for compute resources. We thank Vipul Rathore for his useful feedback on evaluating the quality of the dataset, and Vipul Rathore and Keshav Kolluru for their helpful comments on an earlier draft of the paper.

References

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

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic bert sentence embedding. *arXiv preprint arXiv:2007.01852*.

Xu Han, Tianyu Gao, Yuan Yao, Deming Ye, Zhiyuan Liu, and Maosong Sun. 2019. OpenNRE: An open and extensible toolkit for neural relation extraction. In *Proceedings of EMNLP-IJCNLP: System Demonstrations*, pages 169–174.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 541–550.

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

Sharmistha Jat, Siddhesh Khandelwal, and Partha Talukdar. 2018. Improving distantly supervised relation extraction using word and entity based attention. *arXiv preprint arXiv:1804.06987*.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Abdullatif Köksal and Arzucan Özgür. 2020. The RELX dataset and matching the multilingual blanks for cross-lingual relation classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 340–350, Online. Association for Computational Linguistics.

Keshav Kolluru, Mohammed Muqeeth, Shubham Mittal, Soumen Chakrabarti, and Mausam. 2022. Alignment-Augmented Consistent Translation for Multilingual Open Information Extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, Dublin, Ireland. Association for Computational Linguistics.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2017. Neural relation extraction with multi-lingual attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 34–43.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133.

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

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011.

Jian Ni and Radu Florian. 2019. Neural cross-lingual relation extraction based on bilingual word embedding mapping. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 399–409, Hong Kong, China. Association for Computational Linguistics.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Ji Yoon Han, Jangwon Park, Chisung Song, Junseong Kim, Youngsook Song, Taehwan Oh, Joohong Lee, Juhyun Oh, Sungwon Lyu, Younghoon Jeong, Inkwon Lee, Sangwoo Seo, Dongjun Lee, Hyunwoo Kim, Myeonghwa Lee, Seongbo Jang, Seungwon Do, Sunkyoung Kim, Kyungtae Lim, Jongwon Lee, Kyumin Park, Jamin Shin, Seonghyun Kim, Lucy Park, Alice Oh, Jung-Woo Ha, and Kyunghyun Cho. 2021. KLUE: Korean language understanding evaluation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018. Robust distant supervision relation extraction via deep reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2137–2147. Association for Computational Linguistics.

Vipul Rathore, Kartikeya Badola, Parag Singla, and Mausam. 2022. PARE: A simple and strong baseline for monolingual and multilingual distantly supervised relation extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, Dublin, Ireland. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Alan Ritter, Luke Zettlemoyer, Mausam, and Oren Etzioni. 2013. Modeling missing data in distant supervision for information extraction. *Trans. Assoc. Comput. Linguistics*, 1:367–378.

Alessandro Seganti, Klaudia Firląg, Helena Skowronska, Michał Satława, and Piotr Andruszkiewicz. 2021. Multilingual entity and relation extraction dataset and model. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1946–1955, Online. Association for Computational Linguistics.

Claude E Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Harkanwar Singh, Soumen Chakrabarti, Prachi Jain, Sharod Roy Choudhury, and Mausam. 2021. Multilingual knowledge graph completion with joint relation and entity alignment. In *3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, October 4-8, 2021*.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 455–465.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus.

Xiaozhi Wang, Xu Han, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2018. Adversarial multi-lingual neural relation extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1156–1166.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1753–1762.

Tong Zhu, Haitao Wang, Junjie Yu, Xiabing Zhou, Wenliang Chen, Wei Zhang, and Min Zhang. 2020. Towards accurate and consistent evaluation: A dataset for distantly-supervised relation extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6436–6447, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

A Appendix

B Calculation of Efficiency

For a dataset of size n over k classes, where i^{th} class has n_i instances:

$$Efficiency = -\sum_{i=1}^{k} \frac{\frac{n_i}{n}\log\frac{n_i}{n}}{\log k}$$

Efficiency lies between 0 and 1. A higher value suggests that the class-distribution is closer to uniform.

C Baseline architecture

C.1 BERT Encoder

To obtain a distributed representation of a sentence x, we use mBERT. In order to encode positional information into the model we use Entity Markers scheme introduced by (Soares et al., 2019). We add special tokens $[E1]$, $[\backslash E1]$ to mark start and end of the head entity and $[E2]$, $[\backslash E2]$ to mark start and end of the tail entity. This modified sentence is fed into a pretrained BERT model and the output head and tail tokens are concatenated to get the final sentence representation $\tilde{\mathbf{x}}_i^j$ for each sentence x_i^j in our bag.

C.2 Intra Bag Attention

To obtain representation of bag B, we apply selective sentence-level attention (Lin et al., 2016). We obtain real-valued vector $\tilde{\mathbf{B}}$ for the bag as a weighted sum of sentence representations $\tilde{\mathbf{x}}_i^j$:

$$\tilde{\mathbf{B}} = \sum_{i,j} \alpha_i^j * \tilde{\mathbf{x}}_i^j$$

where α_i^j measures attention score of $\tilde{\mathbf{x}}_i^j$ with a specific relation $\mathbf{r}$:-

$$\alpha_i^j = \frac{exp(\tilde{\mathbf{x}}_i^j \cdot \mathbf{r})}{\sum_{k,l} exp(\tilde{\mathbf{x}}_l^k \cdot \mathbf{r})}$$

This reduces the effect of noisy labels on the final bag representation.

Finally, we obtain conditional probability $p(r|B, \theta) = softmax(\mathbf{o})$. Here we obtain $\mathbf{o}$ which represents scores for all relation types.

$$\mathbf{o} = \mathbf{R}\tilde{\mathbf{B}} + \mathbf{d}$$

$\mathbf{R}$ is the matrix of relation representations. Our objective function is the cross-entropy loss and is defined as follows :-

$$L(\theta) = \sum_{i=1}^{b} p(r_i|B_i, \theta)$$

where b denotes the number of bags in our training data

C.3 MNRE and Cross-Lingual Attention

In order to extend the Intra Bag Attention to multilingual setting, (Lin et al., 2017) introduce separate relation embeddings for each language and propose creating several representations of a bag by taking attention of sentences in language j with relation embedding of language k. Formally, the cross-lingual representation $\mathbf{S}_{jk}$ is defined as a weighted sum of those sentence vectors $\tilde{\mathbf{x}}_i^j$ in the j_{th} language where α_{jk}^i is the attention score of each sentence with respect to the k^{th} language.

$$\mathbf{S}_{jk} = \sum_i \alpha_{ik}^j * \tilde{\mathbf{x}}_i^j$$

$$\alpha_{ik}^j = \frac{exp(\tilde{\mathbf{x}}_i^j \cdot \mathbf{r}_k)}{\sum_l exp(\tilde{\mathbf{x}}_l^j \cdot \mathbf{r}_k)}$$

$$\mathbf{o} = (\mathbf{R}_k + \mathbf{M})\mathbf{S}_{jk} + \mathbf{d}$$

$\mathbf{R}_k$ is the matrix of relation representations ($\mathbf{r}_k$) in language k and $\mathbf{M}$ is a global relation matrix initialized randomly. Similar to (Lin et al., 2016), probability $p(r|\mathbf{S}_{jk}, \theta) = softmax(\mathbf{o})$. To obtain score of relation r for bag B :

$$f(B, r) = \sum_{jk} \log p(r|\mathbf{S}_{j,k}, \theta)$$

Loss function is negative log likelihood over all bags in the dataset.

Language	DiS-ReX (PCNN+Att)		DiS-ReX (mBERT+Att)		DiS-ReX (mBERT+MNRE)	
	AUC	Micro F1	AUC	Micro F1	AUC	Micro F1
English	0.687	0.642	0.781	0.713	0.796	0.733
French	0.714	0.662	0.814	0.746	0.822	0.760
Spanish	0.697	0.644	0.799	0.729	0.816	0.751
German	0.614	0.588	0.757	0.716	0.755	0.717
All languages	0.678	0.634	0.806	0.741	0.817	0.759

Table 7: Language-wise AUC and Micro F1 for baseline models on DiS-ReX

D Training details

For training we use AdamW optimizer (Kingma and Ba, 2017; Loshchilov and Hutter, 2019), with lr=0.001, betas=(0.9, 0.999), eps=1e-08. Weight decay is 0.01 for all parameters except bias and layer norm parameters. Hyperparameters were selected using manual tuning on the dataset. We train the mBERT models for 5 epochs and the PCNN+Att model for 60 epochs. We follow the framework of OpenNRE (Han et al., 2019) and select bag size = 2 for all models. For testing, we choose the weights with best validation AUC. Correct prediction of NA class is not counted in the calculation of Micro F1 and AUC. We use a single Tesla V100 32 GB GPU for all of our experiments.

mBERT+MNRE baseline takes 8 hours for 1 epoch. mBERT+Att takes 3 hours for 1 epoch. PCNN+Att takes 3 hours for 60 epochs.

Training, validation and testing splits for both DiS-ReX and RELX-Distant are in the ratio of 7:1:2. We made sure that the bags in testing set do not overlap with the bags in the training set.

E Detailed Statistics of mBERT Baselines

In Table 7, we present results on all langauges for our three baselines on DiS-ReX. In tables 8, 9 , we present the distribution of errors made by the mBERT+Att and mBERT+MNRE models

In Table 10 and 11, we present the results on bags having 1,2,3 and 4 labels in ground truth using mBERT+Att and mBERT+MNRE respectively.

In Table 12, we present the results on all classes of the best baseline model (mBERT+MNRE) when run on our DiS-ReX dataset.

Language	Type-1 Error (%)	Type-2 Error (%)	Type-3 Error (%)
English	43.44	26.66	29.90
French	29.73	30.45	39.82
Spanish	33.82	30.61	35.57
German	15.03	39.60	45.37

Table 8: Types of Errors made in different languages for mBERT+Att

Language	Type-1 Error (%)	Type-2 Error (%)	Type-3 Error (%)
English	44.49	31.17	24.33
French	29.69	36.14	34.15
Spanish	35.08	36.37	28.54
German	14.94	45.28	39.77

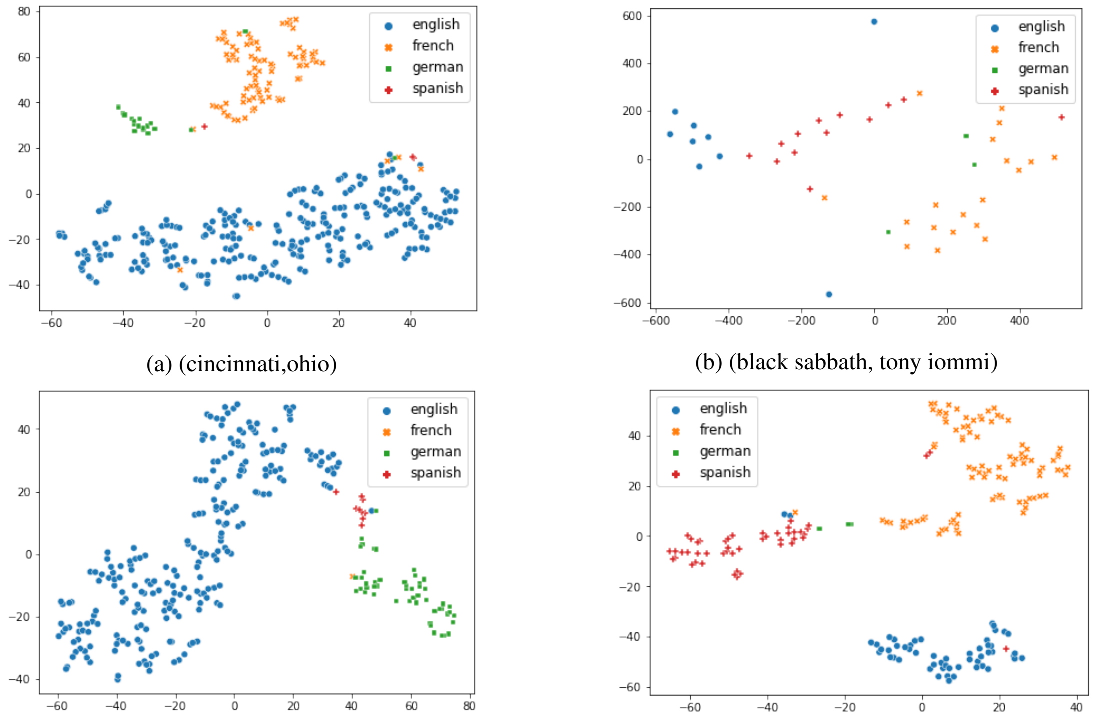
Table 9: Types of Errors made in different languages for mBERT+MNRE

Number of relation labels	Micro-F1	Precision	Recall
1	0.836	0.846	0.825
2	0.662	0.912	0.520
3	0.500	0.939	0.341
4	0.449	0.846	0.305

Table 10: Comparing performance of mBERT+Att on entity pairs with different number of labels in the ground truth

Number of relation labels	Micro-F1	Precision	Recall
1	0.842	0.865	0.820
2	0.673	0.934	0.525
3	0.518	0.959	0.354
4	0.348	0.937	0.214

Table 11: Comparing performance of mBERT+MNRE on entity pairs with different number of labels in the ground truth



(a) (cincinnati,ohio)

(b) (black sabbath, tony iommi)

(c) (miami,florida)

(d) (sumatra, indonesia)

Figure 2: tSNE plot of a few multilingual bags. Languages are marked with different colours

F Some more examples of tSNE plots for mBERT+Att

In figure 2, we provide some more example of tSNE plots for multilingual bags.

We take the following bags:

(cincinnati, ohio) ; (black sabbath, tony iommi)
(miami, florida) ; (sumatra, indonesia)

We use sklearn implementation of tSNE and set the perplexity to be 5.

Relation Label	F1	Precision	Recall
predecessor	67.58	76.31	60.65
nationality	67.29	64.68	70.12
artist	76.78	74.79	78.87
region	81.43	81.14	81.73
department	95.08	95.28	94.88
successor	72.16	75.32	69.26
location	69.82	65.36	74.93
bandMember	73.45	73.45	73.45
isPartOf	66.50	59.52	75.33
hometown	73.03	70.14	76.17
previousWork	68.83	64.89	73.27
riverMouth	72.63	78.97	67.24
team	81.66	85.85	77.86
recordLabel	86.85	87.24	86.46
associatedBand	71.26	61.69	84.36
author	78.87	83.30	74.88
influenced	61.35	65.81	57.46
birthPlace	75.00	75.52	74.48
formerBandMember	57.94	59.62	56.36
leaderName	71.16	70.97	71.35
deathPlace	66.24	64.15	68.46
city	78.96	81.93	76.19
province	78.82	78.73	78.92
influencedBy	59.29	65.26	54.32
locationCountry	62.58	64.76	60.55
related	75.94	74.35	77.59
director	83.59	79.36	88.29
capital	53.68	48.69	59.82
largestCity	65.89	71.57	61.04
NA	95.08	95.56	94.61
country	86.57	85.77	87.39
starring	86.32	86.52	86.12
subsequentWork	71.65	70.23	73.12
producer	53.30	51.20	55.58
headquarter	68.54	66.08	71.18
state	82.54	78.32	87.26
locatedInArea	72.23	70.44	74.10
All relations	70.67	-	-

Table 12: Class-wise performance scores for MNRE (our best performing model)

G Qualitative Analysis

In this section, we give some examples of randomly selected non NA instances in our dataset:
English:

- ***Sentence:*** *another dialect spoken in tioman island is a distinct malay variant and most closely related to riau archipelago malay subdialect spoken in natuna and anambas islands in the south china sea together forming a dialect continuum between the bornean malay with the mainland malay*
  Entities: *(tioman island, the south china sea)*

Relations: *http://dbpedia.org/ontology/location*

- ***Sentence:*** *in 2017 jenny durkan was elected as the first openly lesbian mayor of seattle*
  Entities: *(jenny durkan, seattle)*
  Relations: *http://dbpedia.org/ontology/birthPlace*

German:

- ***Sentence:*** *danach kamen abgeleitete klassen hinzu ein strengeres typsystem und während stroustrup "c with classes" ("c mit klassen") entwickelte woraus später c++ wurde schrieb er auch cfront einen compiler der aus c with classes zunächst c-code als erzeugte*
  Entities: *(c,c++)*
  Relations: *http://dbpedia.org/ontology/influenced*

- ***Sentence:*** *früher auch ur ist ein 96.1 km langer nebenfluss der sauer entlang der grenze von deutschland zu den westlichen nachbarstaaten belgien und luxemburg*
  Entities: *(sauer, deutschland)*
  Relations: *http://dbpedia.org/ontology/locatedInArea*

French:

- ***Sentence:*** *à la mort de boleslas v le pudique duc princeps de pologne la guerre civile en mazovie empêche conrad de revendiquer le trône de cracovie*
  Entities: *(boleslas v le pudique, cracovie)*
  Relations: *http://dbpedia.org/ontology/deathPlace*

- ***Sentence:*** *les entreprises masson masson est le dirigeant effectif des trois entreprises du groupe cette situation se reflète désormais dans l actionnariat et les raisons sociales des sociétés qui deviennent joseph masson sons and company (montréal) masson langevin sons and company (québec) masson sons and company (glasgow) cette dernière société basée en écosse a surtout vocation de gérer les achats*
  Entities: *(joseph masson, québec)*
  Relations: *http://dbpedia.org/ontology/birthPlace*

Spanish:

- ***Sentence:*** *en 2003 apareció en anything else película de woody allen junto a christina ricci y jason biggs además actuó en la película para televisión l*
  Entities: *(anything else, jason biggs)*
  Relations: *http://dbpedia.org/ontology/starring*

- ***Sentence:*** *es una comuna y población de francia en la región de borgoña departamento de yonne en el distrito de sens y cantón de sens-ouest*
  Entities: *(sens, yonne)*
  Relations: *http://dbpedia.org/ontology/department*

H Additional Dataset Statistics

In Table 13, we present the number of bags common across 2,3 and all 4 languages. In table 14 and 15, we present the number of bags and sentences in each class on all 4 languages in our dataset. In figure 3 we present a histogram depicting number of bags present for each relation class.

Number of languages	Number of Bags
2	59709
3	9494
4	1488

Table 13: Number of bags common across 2,3 and all languages

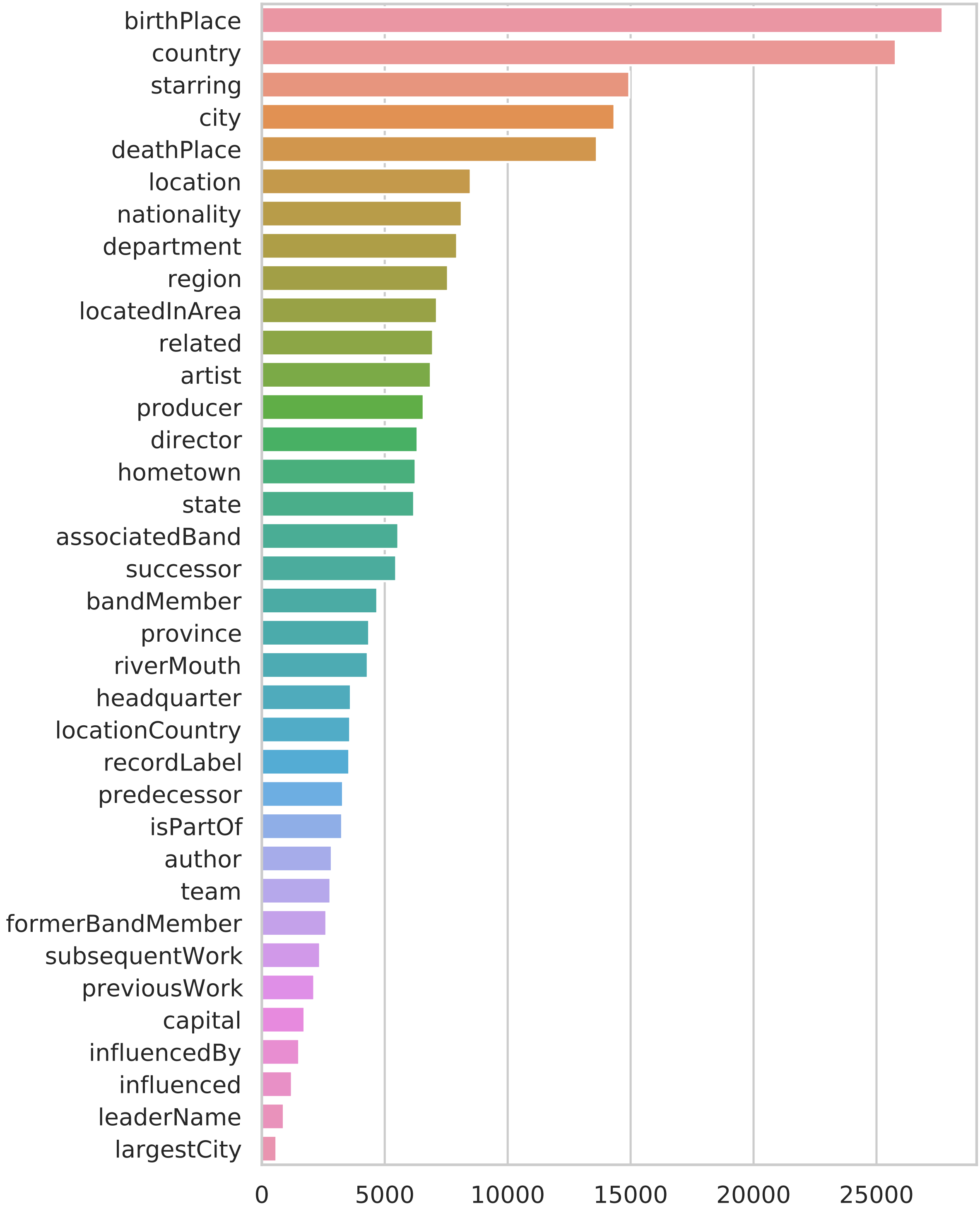


Figure 3: Number of bags vs relation class in DiS-ReX (all languages combined)

Relation Label	English	French	German	Spanish	All languages
NA	149874	142467	149034	148806	590181
isPartOf	2548	645	465	490	4148
state	1882	1762	3537	429	7610
largestCity	265	342	199	393	1199
birthPlace	7861	9532	3341	9484	30218
deathPlace	4377	5629	277	4709	14992
nationality	2205	4413	143	2265	9026
country	10024	9618	3065	9808	32515
capital	544	651	397	891	2483
city	1415	4257	7930	1844	15446
author	1483	1224	94	460	3261
previousWork	348	696	305	1127	2476
location	5655	1300	1180	1685	9820
riverMouth	464	880	3303	154	4801
locatedInArea	1324	785	5715	608	8432
hometown	1689	435	163	4474	6761
successor	1574	2959	74	1618	6225
influenced	820	453	61	188	1522
headquarter	1122	922	460	1895	4399
province	225	1121	1272	2405	5023
associatedBand	3669	384	107	2555	6715
subsequentWork	390	760	344	1248	2742
locationCountry	925	799	2237	361	4322
bandMember	1327	1909	300	3092	6628
director	1258	3003	1592	2089	7942
team	1329	564	461	634	2988
artist	1188	3891	1241	2670	8990
related	1439	375	117	6262	8193
producer	1381	2848	1401	3044	8674
predecessor	475	2814	81	273	3643
leaderName	353	236	270	223	1082
formerBandMember	960	1153	174	1345	3632
recordLabel	791	881	199	2107	3978
region	1529	3673	1907	2249	9358
influencedBy	954	533	86	291	1864
starring	3040	7018	3087	4179	17324
department	99	5486	323	3157	9065
All relations	216806	226418	194942	229512	876743

Table 14: Comprehensive bag-wise statistics of the dataset

Relation Label	English	French	German	Spanish	All languages
NA	231271	167509	278360	224156	901296
isPartOf	16085	2794	2566	1880	23325
state	11979	13135	13705	1405	40224
largestCity	18811	4163	8949	3136	35059
birthPlace	15738	16624	4376	14359	51097
deathPlace	11498	12208	539	8888	33133
nationality	5848	9560	219	4330	19957
country	88787	43911	13148	64660	210506
capital	19887	4713	17227	5318	47145
city	4490	11156	23631	3740	43017
author	3387	4121	335	1417	9260
previousWork	6507	1276	450	2318	10551
location	15538	4757	4656	6014	30965
riverMouth	1172	2442	12467	420	16501
locatedInArea	4320	4152	18890	1904	29266
hometown	7648	796	1067	8971	18482
successor	4700	6963	128	3118	14909
influenced	2416	1147	635	394	4592
headquarter	5419	2399	2030	5736	15584
province	1082	2472	2710	11672	17936
associatedBand	7390	713	136	8437	16676
subsequentWork	6541	1318	517	2526	10902
locationCountry	3204	2836	8226	1229	15495
bandMember	3592	5910	475	8763	18740
director	2005	7811	2970	3961	16747
team	1830	814	694	1396	4734
artist	2893	9591	3156	6472	22112
related	4526	928	171	17432	23057
producer	2459	6398	2647	6384	17888
predecessor	2592	7003	162	600	10357
leaderName	1549	1074	452	448	3523
formerBandMember	2975	3452	279	4091	10797
recordLabel	1320	1214	219	4149	6902
region	5836	11860	5901	4485	28082
influencedBy	2524	1482	913	536	5455
starring	4484	14578	4616	6676	30354
department	196	15807	693	4997	21693
All relations	532499	409087	438315	456418	1858012

Table 15: Comprehensive sentence-wise statistics of the dataset

(Un)solving Morphological Inflection: Lemma Overlap Artificially Inflates Models' Performance

Omer Goldman, David Guriel, Reut Tsarfaty
Bar-Ilan University
{omer.goldman,davidgu1312}@gmail.com,reut.tsarfaty@biu.ac.il

Abstract

In the domain of Morphology, Inflection is a fundamental and important task that gained a lot of traction in recent years, mostly via SIGMORPHON's shared-tasks. With average accuracy above 0.9 over the scores of all languages, the task is considered mostly solved using relatively generic neural seq2seq models, even with little data provided. In this work, we propose to re-evaluate morphological inflection models by employing harder train-test splits that will challenge the generalization capacity of the models. In particular, as opposed to the naïve split-by-form, we propose a split-by-lemma method to challenge the performance on existing benchmarks. Our experiments with the three top-ranked systems on the SIGMORPHON's 2020 shared-task show that the lemma-split presents an average drop of 30 percentage points in macro-average for the 90 languages included. The effect is most significant for low-resourced languages with a drop as high as 95 points, but even high-resourced languages lose about 10 points on average. Our results clearly show that generalizing inflection to unseen lemmas is far from being solved, presenting a simple yet effective means to promote more sophisticated models.

1 Introduction

In recent years, morphological (re)inflection tasks in NLP have gained a lot of attention, most notably with the introduction of SIGMORPHON's shared tasks (Cotterell et al., 2016, 2017, 2018; Vylomova et al., 2020) in tandem with the expansion of UniMorph (McCarthy et al., 2020), a multi-lingual dataset of inflection tables. The shared-tasks sample data from UniMorph includes lists of triplets in the form of *(lemma, features, form)* for many languages, and the shared-task organizers maintain standard splits for a fair system comparison.

The best-performing systems to-date in all inflection shared-tasks are neural sequence-to-sequence models used in many NLP tasks. An LSTM-based model won 2016's task (Kann and Schütze, 2016), and a transformer came on top in 2020 (Canby et al., 2020). In 2020's task the best model achieved exact-match accuracy that transcended 0.9 macro-averaged over up to 90 languages from various language families and types. This trend of high results recurred in works done on data collected independently as well (e.g. Malouf, 2017, Silfverberg and Hulden, 2018, inter alia).

Interestingly, the averaged results of 2020's shared-task include languages for which very little data was provided, sometimes as little as a couple of hundreds of examples. This has led to a view considering morphological inflection a relatively simple task that is essentially already *solved*, as reflected in the saturation of the results over the year and the declining submissions to the shared tasks.[1] This also led the community to gravitate towards works attempting to solve the same (re)inflection tasks with little or no supervision (McCarthy et al., 2019; Jin et al., 2020; Goldman and Tsarfaty, 2021).

However, before moving on we should ask ourselves whether morphological inflection is indeed *solved* or may the good performance be attributed to some artifacts in the data. This was shown to be true for many NLP tasks in which slight modifications of the data can result in a more challenging dataset, e.g., the addition of unanswerable questions to question answering benchmarks (Rajpurkar et al., 2018), or the addition of expert-annotated minimal pairs to a variety of tasks (Gardner et al., 2020). A common modification is re-splitting the data such that the test set is more challenging and closer to the intended use of the models in the wild (Søgaard et al., 2021). As the performance on morphological inflection models seems to have saturated on high scores, a similar rethinking of the data used is warranted.

[1] The shared task of 2021 had seen only two submissions (Pimentel et al., 2021).

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 864 - 870

In this work we propose to construct more difficult datasets for morphological (re)inflection by splitting them such that the test set will include no forms of lemmas appearing in the train set. This splitting method will allow assessing the models in a challenging scenario closer to their desired function in practice, where training data usually includes full inflection tables and learning to inflect the uncovered lemmas is the target.

We show, by re-splitting the data from task 0 of SIGMORPHON's 2020 shared-task, that the proposed split reveals a greater difficulty of morphological inflection. Retesting 3 of the 4 top-ranked systems of the shared-task on the new splits leads to a decrease of 30 points averaged over the systems for all 90 languages included in the shared-task. We further show that the effect is more prominent for low-resourced languages, where the drop can be as large as 95 points, though high-resourced languages may suffer from up to a 10 points drop as well. We conclude that in order to properly assess the performance of (re)inflection models and to drive the field forward, the data and related splits should be carefully examined and improved to provide a more challenging evaluation, more reflective of their real-world use.

2 (Re)inflection and Memorization

Inflection and reinflection are two of the most dominant tasks in computational morphology. In the *inflection* task, the input is a lemma and a feature-bundle, and we aim to predict the respective inflected word-form. In *reinflection*, the input is an inflected word-form along with its features bundle, plus a feature-bundle without a form, and we aim to predict the respective inflected-form for the same lemma. The *training* input in SIGMORPHON's shared-tasks is a random split of the available *(lemma,form,features)* triplets such that no triplet occurring in the train-set occurs in the test-set.[2]

In such a setting, models can short-cut their way to better predictions in cases where forms from the same lemma appear in both the train and test data. This may allow models to memorize lemma-specific alternations that make morphological inflection a challenging task to begin with. Consider for example the notoriously unpredictable German plurality marking, where several allomorphs are associated with nouns with no clear rule governing the process. *Kind*, for example, is pluralized with the suffix *-er* resulting in *Kinder* tagged as NOM;PL. Assuming a model saw this example in the train set it is pretty easy to predict *Kindern* for the same lemma with DAT;PL features,[3] but without knowledge of the suffix used to pluralize *Kind* the predictions *Kinden* and *Kinds* are just as likely.

3 Related Work

Many subfields of NLP and machine learning in general suggested *hard splits* as means to improve the probing of models' ability to solve the underlying task, and to make sure models do not simply employ loopholes in the data.

In the realm of sentence simplification, Narayan et al. (2017) suggested the WEBSPLIT dataset, where models are required to split and rephrase complex sentences associated with a meaning representation over a knowledge-base. Aharoni and Goldberg (2018) found that some facts appeared in both train and test sets and provided a harder split denying models the ability to use memorized facts. Aharoni and Goldberg (2020) also suggested a general splitting method for machine translation such that the domains are as disjoint as possible.

In semantic parsing, Finegan-Dollak et al. (2018) suggested a better split for parsing natural language questions to SQL queries by making sure that queries of the same template do not occur in both train and test, while Lachmy et al. (2021) split their HEXAGONS data such that any one visual pattern used for the task cannot appear in both train and test. Furthermore, Loula et al. (2018) adversarially split semantic parsing for navigation data to assess their models' capability to use compositionality. In spoken language understanding Arora et al. (2021) designed a splitting method that will account for variation in both speaker identity and linguistic content.

In general, concerns regarding data splits and their undesired influence on model assessments led Gorman and Bedrick (2019) to advocate random splitting instead of standard ones. In reaction, Søgaard et al. (2021) pointed to the flaws of random splits and suggested adversarial splits to challenge models further. Here we call for paying attention to the splits employed in evaluating morphological models, and improve on them.

[2] This is true for all SIGMORPHON's inflection shared tasks, save the paradigm completion task of 2017.

[3] The addition of the dative marker *-n* is very regular.

	Accuracy		Edit Distance	
Split	**Form**	**Lemma**	**Form**	**Lemma**
DeepSpin-02	**0.90**	**0.76**	**0.23**	**0.58**
CULing	0.88	0.63	0.29	1.02
Base trm-single	**0.90**	0.53	**0.23**	1.32
Base LSTM	0.85	0.39	0.34	1.79
Average	0.88	0.58	0.27	1.18

Table 1: Exact-match accuracy and edit-distance for our baseline and 3 of the 4 top-ranked systems of SIGMORPHON's 2020 shared-task, all reported on the original split of the shared-task (form split) and on our harder lemma split. Best system per column is in **bold**.

	Accuracy	
Split	**Form**	**Lemma**
Afro-Asiatic	0.93 (0.95)$_{\text{T}}$	0.51 (0.80)$_{\text{D}}$
Austronesian	0.78 (0.82)$_{\text{T}}$	0.45 (0.70)$_{\text{D}}$
Germanic	0.86 (0.88)$_{\text{D}}$	0.63 (0.74)$_{\text{D}}$
Indo-Iranian	0.93 (0.97)$_{\text{D}}$	0.55 (0.86)$_{\text{D}}$
Niger-Congo	0.95 (0.98)$_{\text{T}}$	0.56 (0.90)$_{\text{D}}$
Oto-Manguean	0.84 (0.86)$_{\text{T}}$	0.53 (0.60)$_{\text{D}}$
Romance	0.97 (0.99)$_{\text{T}}$	0.69 (0.86)$_{\text{D}}$
Turkic	0.95 (0.96)$_{\text{T}}$	0.64 (0.89)$_{\text{D}}$
Uralic	0.88 (0.90)$_{\text{C}}$	0.65 (0.72)$_{\text{D}}$

Table 2: Aggregated results for the various language families. We provide the performance averaged across all systems, and in parenthesis the performance of the best system per family. The best system is identifiable in subscript: C - CULing, T - Base trm-single, D - DeepSpin-02. We include here only families with at least 3 languages in the data.

4 Experiments

In order to better assess the difficulty of morphological inflection, we compare the performances of 3 of the top-ranked system at task 0 (inflection) of SIGMORPHON's 2020 shared-task. We examined each system on both the the standard (form) split and the novel (lemma) split.

When re-splitting,[4] we kept the same proportions of the form-split data, i.e. we split the inflection tables 70%, 10% and 20% for the train, development and test set. In terms of examples the proportions may vary as not all tables are of equal size. In practice, the averaged train set size in examples terms was only 3.5% smaller in the lemma-split data, on average.[5]

4.1 The Languages

SIGMORHPON's 2020 shared-task includes datasets for 90 typologically and genealogically diverse languages from 14 language families. The languages are varied along almost any typological dimension, from fusional to agglutinative, small inflection tables to vast ones. They include mostly prefixing and mostly suffixing languages with representation of infixing and circumfixing as well. The languages vary also in use, including widely-used languages such as English and Hindi and moribund or extinct languages like Dakota and Middle High German.[6]

4.2 The Models

We tested the effects of lemma-splitting on our own LSTM-based model as well as 3 of the 4 top-ranked systems in the shared task.[7]

Base LSTM We implemented a character-based sequence-to-sequence model which consists of a 1-layer bi-directional LSTM Encoder and a 1-layer unidirectional LSTM Decoder with a global soft attention layer (Bahdanau et al., 2014). Our model was trained for 50 epochs with no model selection.[8]

Base trm-single The shared-task's organizers supplied various baselines, some based on a transformer architecture that was adapted for character-level tasks (Wu et al., 2021).[9] All baseline models include 4 encoder and 4 decoder layers, consisting of a multi-head self-attention layer and 2 feed-forward layers, equipped with a skip-connection. In every decoder layer a multi-head attention layer attends to the encoder's outputs. The network was trained for 4,000 warm-up steps and up to 20,000 more steps, each step over a batch of size 400. The model was examined with and without augmented data, trained separately on each language or each language family. One of the baseline setups, training a model per language without augmented data, made it to the top 4 systems and we include it here.

[4] The split was done randomly as is standard in SIGMORPHON tasks, although frequency-based sampling is also conceivable and is sometimes used, as in Cotterell et al. (2018).

[5] The newly-split data is available at `https://github.com/OnlpLab/LemmaSplitting`.

[6] The full list with the originally released data are at `https://github.com/sigmorphon2020/task0-data`.

[7] The best performing system, UIUC (Canby et al., 2020), did not have a publicly available implementation.

[8] The code is available at `https://github.com/OnlpLab/LemmaSplitting`.

[9] The code is available at `https://github.com/shijie-wu/neural-transducer`.

DeepSpin Peters and Martins (2020) submitted a recurrent neural network – dubbed DeepSpin-02.[10] The system is composed of 2 bi-directional LSTM encoders with bi-linear gated Attention (Luong et al., 2015), one for the lemma characters and one for the features characters, and a unidirectional LSTM Decoder for generating the outputs. The innovation in the architecture is the use of sparsemax (Martins and Astudillo, 2016) instead of softmax in the attention layer.[11]

CULing Liu and Hulden (2020)'s system is also based on the transformer architecture, with hyper-parameters very similar to *base trm-single*.[12] Their innovation is in restructuring the data such that the model learns to inflect from any given cell in the inflection table rather than solely from the lemma.

4.3 Results

Table 1 summarizes our main results. We clearly see a drop in the performance for all systems, with an average of 30 points. The table also shows that splitting the data according to lemmas allows discerning between systems that appear to perform quite similarly on the form-split data. The best system on the lemma-split data, DeepSpin-02, outperforms the second-ranked CULing system by about 13 points with both baseline systems performing significantly worse. The results in terms of averaged edit distance show the same trends.

DeepSpin-02 emerges victorious also in Table 2, where results are broken down by language family. The table shows that DeepSpin-02 is the best performer over all language families when data is lemma-split, in contrast to the mixed picture over the form-split data.

The average performance per language family seems to be controlled by training data availability. For example, Germanic languages show average drop of 23 points, while for Niger-Congo languages the drop is 39 points on average.

In order to further examine the relation between the amount of training data and drop in performance we plotted in Figure 1 the drop per system and per language against the size of the available train data, color-coded to indicate systems. It shows that the major drops in performance that contributed the most to the overall gap between the splits are in those low-resourced language. Remarkably, for some systems and languages the drop can be as high as 95 points. On the other hand, on high-resourced languages with 40,000 training examples or more, all systems didn't lose much. The analysis also shows the advantage of DeepSpin-02 in the lower-resourced settings that made it the best performer overall.

When color-coding the same broken-down data for linguistic family membership rather than system, as we do in Figure 2, it becomes clear that there is no evidence for specific families being eas-

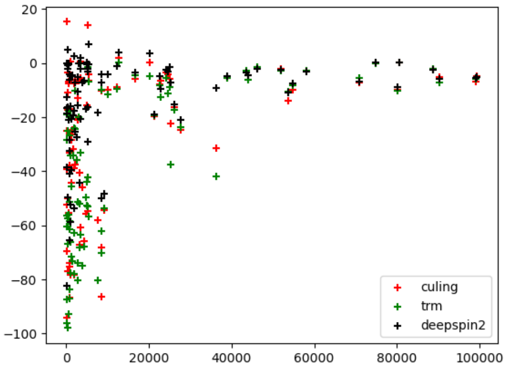


Figure 1: Performance drop for the various systems when moving from form to lemma split as a function of the size of the train data. The effect is clearly more significant for lower-resourced languages.

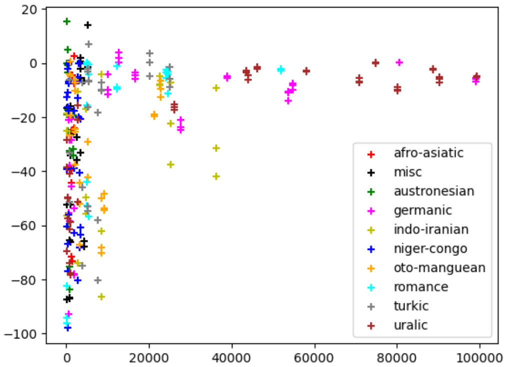


Figure 2: Performance drop for the various language families when moving from form to lemma split as a function of the size of the train data. We include here only families with at least 3 languages in the data, the rest are classified under *misc*.

[10] The code is available at https://github.com/deep-spin/sigmorphon-seq2seq.

[11] The system submitted as DeepSpin-01 uses 1.5-entmax (Peters and Martins, 2019) rather than sparsemax. Both systems perform highly similarly, hence we do not detail results for both.

[12] The code is available at https://github.com/LINGuistLIU/principal_parts_for_inflection.

ier for inflection when little data is provided. The figure does show the remarkable discrepancy in annotation effort, as the high-resourced languages mostly belong to 2 families: Germanic and Uralic.

5 Discussion

We proposed a method for splitting morphological datasets such that there is no lemma overlap between the splits. On the re-split of SIGMORPHON's 2020 shared-task data, we showed that all top-ranked systems suffer significant drops in performance. The new split examines models' generalization abilities in conditions more similar to their desired usage in the wild and allows better discerning between the systems in order to point to more promising directions for future research — more so than the original form-split data on which all systems fared similarly. The new splitting method is likely to lead to more sophisticated modeling, for instance, in the spirit of the model proposed by Liu and Hulden (2021). The suggested move to a harder split is not unlike many other NLP tasks, in which challenging splits are suggested to drive the field forward. We thus call for morphological studies to carefully attend to the data used and expose the actual difficulties in modelling morphology, in future research and future shared tasks.

Acknowledgements

This research was funded by the European Research Council under the European Union's Horizon 2020 research and innovation programme, (grant agreement No. 677352) and by a research grant from the ministry of Science and Technology (MOST) of the Israeli Government, for which we are grateful.

References

Roee Aharoni and Yoav Goldberg. 2018. Split and rephrase: Better evaluation and stronger baselines. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 719–724, Melbourne, Australia. Association for Computational Linguistics.

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763, Online. Association for Computational Linguistics.

Siddhant Arora, Alissa Ostapenko, Vijay Viswanathan, Siddharth Dalmia, Florian Metze, Shinji Watanabe, and Alan W Black. 2021. Rethinking end-to-end evaluation of decomposable tasks: A case study on spoken language understanding.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. Cite arxiv:1409.0473Comment: Accepted at ICLR 2015 as oral presentation.

Marc Canby, Aidana Karipbayeva, Bryan Lunt, Sahand Mozaffari, Charlotte Yoder, and Julia Hockenmaier. 2020. University of Illinois submission to the SIGMORPHON 2020 shared task 0: Typologically diverse morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 137–145, Online. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017. CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 1–30, Vancouver. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The SIGMORPHON 2016 shared Task—Morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22, Berlin, Germany. Association for Computational Linguistics.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia. Association for Computational Linguistics.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

Omer Goldman and Reut Tsarfaty. 2021. Minimal supervision for morphological inflection.

Kyle Gorman and Steven Bedrick. 2019. We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2786–2791, Florence, Italy. Association for Computational Linguistics.

Huiming Jin, Liwei Cai, Yihui Peng, Chen Xia, Arya McCarthy, and Katharina Kann. 2020. Unsupervised morphological paradigm completion. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6696–6707, Online. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2016. MED: The LMU system for the SIGMORPHON 2016 shared task on morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 62–70, Berlin, Germany. Association for Computational Linguistics.

Royi Lachmy, Valentina Pyatkin, and Reut Tsarfaty. 2021. Draw me a flower: Grounding formal abstract structures stated in informal natural language.

Ling Liu and Mans Hulden. 2020. Leveraging principal parts for morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 153–161, Online. Association for Computational Linguistics.

Ling Liu and Mans Hulden. 2021. Can a transformer pass the wug test? tuning copying bias in neural morphological inflection models. *CoRR*, abs/2104.06483.

João Loula, Marco Baroni, and Brenden Lake. 2018. Rearranging the familiar: Testing compositional generalization in recurrent networks. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 108–114, Brussels, Belgium. Association for Computational Linguistics.

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

Robert Malouf. 2017. Abstractive morphological learning with a recurrent neural network. *Morphology*, 27(4):431–458.

André F. T. Martins and Ramón F. Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1614–1623. JMLR.org.

Arya D. McCarthy, Christo Kirov, Matteo Grella, Amrit Nidhi, Patrick Xia, Kyle Gorman, Ekaterina Vylomova, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, Timofey Arkhangelskiy, Nataly Krizhanovsky, Andrew Krizhanovsky, Elena Klyachko, Alexey Sorokin, John Mansfield, Valts Ernštreits, Yuval Pinter, Cassandra L. Jacobs, Ryan Cotterell, Mans Hulden, and David Yarowsky. 2020. UniMorph 3.0: Universal Morphology. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3922–3931, Marseille, France. European Language Resources Association.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sabrina J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy. Association for Computational Linguistics.

Shashi Narayan, Claire Gardent, Shay B. Cohen, and Anastasia Shimorina. 2017. Split and rephrase. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 606–616, Copenhagen, Denmark. Association for Computational Linguistics.

Ben Peters and André F. T. Martins. 2019. IT–IST at the SIGMORPHON 2019 shared task: Sparse two-headed models for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 50–56, Florence, Italy. Association for Computational Linguistics.

Ben Peters and André F. T. Martins. 2020. One-size-fits-all multilingual models. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 63–69, Online. Association for Computational Linguistics.

Tiago Pimentel, Maria Ryskina, Sabrina J. Mielke, Shijie Wu, Eleanor Chodroff, Brian Leonard, Garrett Nicolai, Yustinus Ghanggo Ate, Salam Khalifa, Nizar Habash, Charbel El-Khaissi, Omer Goldman, Michael Gasser, William Lane, Matt Coler, Arturo Oncevay, Jaime Rafael Montoya Samame, Gema Celeste Silva Villegas, Adam Ek, Jean-Philippe Bernardy, Andrey Shcherbakov, Aziyana Bayyr-ool, Karina Sheifer, Sofya Ganieva, Matvey Plugaryov, Elena Klyachko, Ali Salehi, Andrew Krizhanovsky, Natalia Krizhanovsky, Clara Vania, Sardana Ivanova, Aelita Salchak, Christopher Straughn, Zoey Liu, Jonathan North Washington, Duygu Ataman, Witold Kieraś, Marcin Woliński, Totok Suhardijanto, Niklas Stoehr, Zahroh Nuriah, Shyam Ratan, Francis M. Tyers, Edoardo M. Ponti, Grant Aiton, Richard J. Hatcher, Emily Prud'hommeaux, Ritesh Kumar, Mans Hulden, Botond Barta, Dorina Lakatos, Gábor Szolnok, Judit Ács, Mohit Raj, David Yarowsky, Ryan Cotterell, Ben Ambridge, and Ekaterina Vylomova. 2021. Sigmorphon 2021 shared task on morphological reinflection: Generalization across languages. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–259, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Miikka Silfverberg and Mans Hulden. 2018. An encoder-decoder approach to the paradigm cell filling problem. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2883–2889, Brussels, Belgium. Association for Computational Linguistics.

Anders Søgaard, Sebastian Ebert, Jasmijn Bastings, and Katja Filippova. 2021. We need to talk about random splits. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1823–1832, Online. Association for Computational Linguistics.

Ekaterina Vylomova, Jennifer White, Elizabeth Salesky, Sabrina J. Mielke, Shijie Wu, Edoardo Maria Ponti, Rowan Hall Maudslay, Ran Zmigrod, Josef Valvoda, Svetlana Toldova, Francis Tyers, Elena Klyachko, Ilya Yegorov, Natalia Krizhanovsky, Paula Czarnowska, Irene Nikkarinen, Andrew Krizhanovsky, Tiago Pimentel, Lucas Torroba Hennigen, Christo Kirov, Garrett Nicolai, Adina Williams, Antonios Anastasopoulos, Hilaria Cruz, Eleanor Chodroff, Ryan Cotterell, Miikka Silfverberg, and Mans Hulden. 2020. SIGMORPHON 2020 shared task 0: Typologically diverse morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 1–39, Online. Association for Computational Linguistics.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2021. Applying the transformer to character-level transduction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1901–1907, Online. Association for Computational Linguistics.

Text Smoothing: Enhance Various Data Augmentation Methods on Text Classification Tasks

Xing Wu[1,2,3], **Chaochen Gao**[1,2]*, **Meng Lin**[1,2], **Liangjun Zang**[1], **Songlin Hu**[1,2]†
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]Kuaishou Technology, Beijing, China
{gaochaochen,linmeng,zangliangjun,husonglin}@iie.ac.cn
wuxing@kuaishou.com

Abstract

Before entering the neural network, a token is generally converted to the corresponding one-hot representation, which is a discrete distribution of the vocabulary. Smoothed representation is the probability of candidate tokens obtained from a pre-trained masked language model, which can be seen as a more informative substitution to the one-hot representation. We propose an efficient data augmentation method, termed **text smoothing**, by converting a sentence from its one-hot representation to a controllable smoothed representation. We evaluate text smoothing on different benchmarks in a low-resource regime. Experimental results show that text smoothing outperforms various mainstream data augmentation methods by a substantial margin. Moreover, text smoothing can be combined with those data augmentation methods to achieve better performance. Our code are available at https://github.com/caskcsg/TextSmoothing.

1 Introduction

Data augmentation is a widely used technique, especially in the low-resource regime. It increases the size of the training data to alleviate overfitting and improve the robustness of deep neural networks. In the field of natural language processing (NLP), various data augmentation techniques have been proposed. One most commonly used method is to randomly select tokens in a sentence and replace them with semantically similar tokens to synthesize a new sentence (Wei and Zou, 2019; Kobayashi, 2018). (Kobayashi, 2018) proposes contextual augmentation to predict the probability distribution of replacement tokens by using the LSTM language model and sampling the replacement tokens according to the probability distribution. (Wu et al., 2019a,b) uses BERT's (Devlin et al., 2018) masked language modeling (MLM) task to extend contextual augmentation by considering deep bi-directional context. (Kumar et al., 2020) further propose to use different types of transformer based pre-trained models for conditional data augmentation in the low-resource regime.

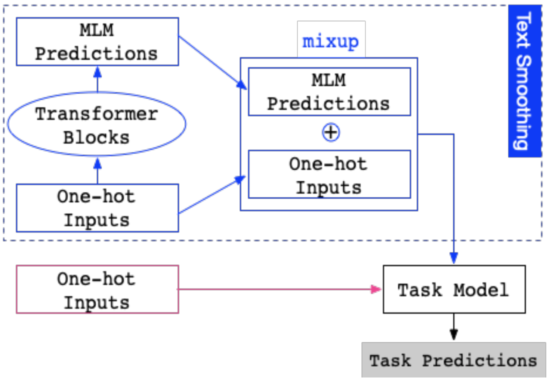


Figure 1: The blue part demonstrates the use of text smoothing data augmentation for downstream tasks, and the red part directly uses the original input.

MLM takes masked sentences as input, and typically 15% of the original tokens in the sentences will be replaced by the [MASK] token. Before entering MLM, each token in sentences needs to be converted to its one-hot representation, a vector of the vocabulary size with only one position is 1 while the rest positions are 0. MLM outputs the probability distribution of the vocabulary size of each mask position. Through large-scale pretraining, it is expected that the probability distribution is as close as possible to the ground-truth one-hot representation. Compared with the one-hot representation, the probability distribution predicted by pre-trained MLM is a "smoothed" representation, which can be seen as a set of candidate tokens with different weights. Usually, most of the weights are distributed on contextual-compatible tokens. Multiplying the smooth representation by the word embedding matrix can obtain a weighted summation of the word embeddings of the candidate words, termed smoothed embedding, which is more informative and context-rich than the one-

*The first two authors contribute equally.
†Corresponding author.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 871 - 875
May 22-27, 2022 ©2022 Association for Computational Linguistics

hot's embedding obtained through lookup operation. Therefore, the use of smoothed representation instead of one-hot representation as the input of the model can be seen as an efficient weighted data augmentation method. To get the smoothed representation of all the tokens of the entire sentence with only one forward process in MLM, we do not explicitly mask the input. Instead, we turn on the dropout of MLM and dynamically randomly discard a portion of the weight and hidden state at each layer.

An unneglectable situation is that some tokens appear more frequently than others in similar contexts during pre-training, which will cause the model to have a preference for these tokens. This is harmful for downstream tasks such as fine-grained sentiment classification. For example, given "The quality of this shirt is average .", the "average" token is most relevant to the label. The smoothed representation through the MLM at the position of "average" is shown in Figure 2. Although the probability of "average" is the highest, more probabilities are concentrated on tokens conflict with the task label, such as "high", "good" or "poor". Such a smoothed representation is hardly a good augmented input for the task. To solve this problem, (Wu et al., 2019a) proposed to train label embedding to constraint MLM predict label compatible tokens. However, under the condition of low resources, it is not easy to have enough label data to provide supervision. Inspired by the practical data augmentation method mixup (Zhang et al., 2017) in the computer vision field, we interpolate the smoothed representation with the original one-hot representation. Through interpolation, we can enlarge the probability of the original token, and the probabilities are still mostly distributed on the context-compatible words, as shown in the figure 2.

We combine the two stages as **text smoothing**: obtaining a smooth representation through MLM and interpolating to constrain the representation more controllable. To evaluate the effect of text smoothing, we perform experiments with low-resource settings on three classification benchmarks. In all experiments, text smoothing achieves better performance than other data augmentation methods. Further, we are pleased to find that text smoothing can be combined with other data augmentation methods to improve the tasks further. To the best of our knowledge, this is the first method to improve a variety of mainstream data augmentation methods.

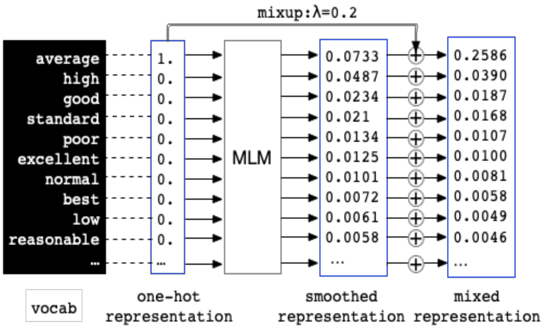


Figure 2: Interpolation of the smoothed representation and the original one-hot representation.

2 Related Work

Various NLP data augmentation techniques have been proposed and they are mainly divided into two categories: one is to modify raw input directly, and the other interferes with the embedding (Miyato et al., 2016; Zhu et al., 2019). The most commonly used method to modify the raw input is the token replacement: randomly select tokens in a sentence and replace them with semantically similar tokens to synthesize a new sentence. (Wei and Zou, 2019) directly uses the synonym table WordNet(Miller, 1998) for replacement. (Kobayashi, 2018) proposes contextual augmentation to predict the probability distribution of replacement tokens with two causal language models. (Wu et al., 2019a) extends contextual augmentation with BERT's masked language modeling (MLM) to consider bi-directional context. (Gao et al., 2019) softly augments a randomly chosen token in a sentence by replacing its one-hot representation with the distribution of the vocabulary provided by the causal language model in machine translation. Unlike (Gao et al., 2019), we use MLM to generate smoothed representation, which considers the deep bi-directional context more adequately. And our method has better parallelism, which can efficiently obtain the smoothed representation of the entire sentence in one forward process. Moreover, we propose to constrain smoothed representation more controllable through interpolation for classification tasks.

3 Our Method

3.1 Smoothed Representation

We use BERT as a representative example of MLM. Given a downstream task dataset, namely $\mathcal{D} = \{t_i, p_i, s_i, l_i\}_{i=1}^{N}$, where N is the number of

```
    sentence = "My favorite fruit is pear ."
    lambd = 0.1 # interpolation hyperparameter
    mlm.train() # enable dropout, dynamically mask
    tensor_input = tokenizer(sentence, return_tensors="pt")
    onehot_repr = convert_to_onehot(**tensor_input)
    smoothed_repr = softmax(mlm(**tensor_input).logits[0])
    interpolated_repr = lambd * onehot_repr + (1 - lambd) * smoothed_repr
```

Listing 1: Codes to implement text smoothing in PyTorch

instances, t_i is the one-hot encoding of a text (a single sentence or a sentence pair), p_i is the positional encoding of t_i, s_i is the segment encoding of t_i and l_i is the label of this instance. We feed the one-hot encoding t_i, positional encoding p_i as well as the segment encoding s_i into BERT, and fetch the output of the last layer of the transformer encoder in BERT, which is denoted as:

$$\overrightarrow{t_i} = \mathrm{BERT}(t_i) \tag{1}$$

where $\overrightarrow{t_i} \in \mathcal{R}^{\text{seq\_len,emb\_size}}$ is a 2D dense vector in shape of [sequence_len, embedding_size]. We then multiply $\overrightarrow{t_i}$ with the word embedding matrix $W \in \mathcal{R}^{\text{vocab\_size,embed\_size}}$ in BERT, to get the MLM prediction results, which is defined as:

$$\mathrm{MLM}(t_i) = \mathrm{softmax}(\overrightarrow{t_i} W^T) \tag{2}$$

where each row in $\mathrm{MLM}(t_i)$ is a probability distribution over the token vocabulary, representing the context-compatible token choices in that position of the input text learned by pre-trained BERT.

3.2 Mixup Strategy

The mixup (Zhang et al., 2017) is defined as:

$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j \tag{3}$$

$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j \tag{4}$$

where (x_i, y_i) and (x_j, y_j) are two feature-target vectors drawn at random from the training data, and $\lambda \in [0, 1]$. In text smoothing, the one-hot representation and smoothed representation are derived from the same raw input, their lables are identical and the interpolation operation will not change the label. So the mixup operation can be simplified to:

$$\widetilde{t_i} = \lambda \cdot t_i + (1 - \lambda) \cdot \mathrm{MLM}(t_i) \tag{5}$$

where t_i is the one-hot representation, $\mathrm{MLM}(t_i)$ is the smoothed representation, $\widetilde{t_i}$ is the interpolated representation and λ is the balance hyperparameter to control interpolation strength. In the downstream tasks, we use interpolated representation instead of the original one-hot representation as input.

	SST-2	SNIPS	TREC
Train	20	70	60
Dev	20	70	60
Test	1821	700	500

Table 1: Data statistics in low-resource regime settings.

4 Experiment

4.1 Baseline Approaches

EDA(Wei and Zou, 2019) consists of four simple operations: synonym replacement, random insertion, random swap, and random deletion.

Back Translation (Shleifer, 2019) translate a sentence to a temporary language (EN-DE) and then translate back the previously translated text into the source language (DE-EN).

CBERT (Wu et al., 2019a) masks some tokens and predicts their contextual substitutions with pre-trained BERT.

BERTexpand, BERTprepend (Kumar et al., 2020) conditions BERT by prepending class labels to all examples of given class. “expand" a the label to model vocabulary, while “prepend" without.

GPT2context (Kumar et al., 2020) provides a prompt to the pre-trained GPT model and keeping generating until the EOS token.

BARTword, BARTspan (Kumar et al., 2020) conditions BART by prepending class labels to all examples of given class. BARTword masks a single word while BARTspan masks a continuous chunk.

4.2 Experiment Setting

Our experiment strictly follows the settings in the (Kumar et al., 2020) paper on three text classification datasets downloaded from the links [1].

SST-2 (Socher et al., 2013) is a movie reviews sentiment classification task with two labels.

SNIPS (Coucke et al., 2018) is a task of over 16,000 crowd-sourced queries distributed among 7 user intents of various complexity.

[1] SST-2 and TREC:https://github.com/1024er/cbert_aug,
SNIPS:https://github.com/MiuLab/SlotGated-SLU/tree/master/data/snips

Method	SST-2	SNIPS	TREC	Avg.
No Aug	52.93 (5.01)	79.38 (3.20)	48.56 (11.53)	60.29(6.58)
EDA	53.82 (4.44)	85.78 (2.96)	52.57 (10.49)	64.06(5.96)
BackTrans.	57.45 (5.56)	86.45 (2.40)	66.16 (8.52)	70.02(5.49)
CBERT	57.36 (6.72)	85.79 (3.46)	64.33 (10.90)	69.16(7.03)
BERTexpand	56.34 (6.48)	86.11 (2.70)	65.33 (6.05)	69.26(5.08)
BERTprepend	56.11 (6.33)	86.77 (1.61)	64.74 (9.61)	69.21(5.85)
GPT2context	55.40 (6.71)	86.59 (2.73)	54.29 (10.12)	65.43(6.52)
BARTword	57.97 (6.80)	86.78 (2.59)	63.73 (9.84)	69.49(6.41)
BARTspan	57.68 (7.06)	87.24 (1.39)	67.30 (6.13)	70.74(4.86)
Text smoothing	**59.37(7.79)**	**88.85(1.49)**	**67.51(7.46)**	**71.91 (5.58)**

Table 2: Evaluating data augmentation methods on different datasets in a low-resource regime.

Method	SST-2	SNIPS	TREC	Avg.
EDA	59.66 (5.57)	87.53 (2.31)	55.95 (7.90)	67.71 (5.26)
+ text smoothing	**64.84(6.82)**	**88.54(3.03)**	**67.68(9.70)**	**73.69(6.52)**
BackTrans.	60.60 (7.40)	86.04 (2.20)	64.57 (7.48)	70.40 (5.70)
+ text smoothing	**61.66(7.62)**	**88.72(1.99)**	**69.17(10.51)**	**73.19(6.7)**
CBERT	60.10 (4.57)	86.85 (2.06)	63.56 (8.09)	70.17 (4.91)
+ text smoothing	**61.65(6.65)**	**88.18(2.85)**	**67.84(9.70)**	**72.56(6.4)**
BERTexpand	59.85 (6.16)	86.12 (2.45)	62.67 (7.59)	69.55 (5.40)
+ text smoothing	**62.04(7.93)**	**89.49(2.05)**	**65.89(7.48)**	**72.47(5.82)**
BERTprepend	60.28 (5.80)	86.86 (2.46)	65.20 (6.88)	70.78 (5.05)
+ text smoothing	**62.75(7.14)**	**88.04(1.92)**	**68.07(7.30)**	**72.95(5.45)**
GPT2context	57.46 (4.96)	84.10 (2.39)	46.47 (12.80)	62.68 (6.72)
+ text smoothing	**60.66(6.72)**	**87.68(1.60)**	**59.13(11.33)**	**69.16(6.55)**
BARTword	60.99(7.15)	86.98(1.96)	61.29(10.00)	69.76(6.37)
+ text smoothing	**62.67(7.40)**	**88.50(2.10)**	**67.75(6.50)**	**72.97(5.33)**
BARTspan	**63.42(5.58)**	87.34(2.17)	62.47(8.11)	71.08(5.29)
+ text smoothing	62.37(7.18)	**89.06(2.18)**	**70.89(6.81)**	**74.11(5.39)**

Table 3: The effect of text smoothing combined with other data augmentation methods in low-resource regime.

TREC (Li and Roth, 2002) contains six question types collected from 4,500 English questions.

We randomly subsample 10 examples per class for each experiment for both training and development set to simulate a low-resource regime. Data statistics of the three datasets are shown in Table 1. Following (Kumar et al., 2020), we replace numeric class labels with their text versions.

We first compare the effects of text smoothing and baselines data augmentation methods on different datasets in a low-resource regime. Then we further explore the effect of combining text smoothing with each baseline method. Considering that the amount of data increases to 2 times after combination, we expand the data used in the baseline experiments to the same amount for the fairness of comparison. All experiments are repeated 15 times to account for stochasticity and results are reported as Mean (STD) accuracy on the full test set.

4.3 Experimental Results

As shown in Table2, text smoothing brings the largest improvement to the model on the three datasets compared with other data augmentation methods. The previously best method is BARTspan, which is exceeded by Text smoothing with 1.17% in average.

Moreover, we are pleased to find that text smoothing can be well combined with various data augmentation methods, further improving the baseline data augmentation methods. As shown in Table3, text smoothing can bring significant improvements of 5.98%, 2.79%, 2.39%, 2.92%, 2.17%, 6.48%, 3.21%, 3.03% to EDA, BackTrans, CBERT, BERTexpand, BERTprepend, GPT2context, BARTword, and BARTspan, respectively. To the best of our knowledge, this is the first method to improve a variety of mainstream data augmentation methods.

5 Conclusoins

This article proposes text smoothing, an effective data augmentation method, by converting sentences from their one-hot representations to smoothing representations. In the case of a low data regime, text smoothing is significantly better than various data augmentation methods. Furthermore, text smoothing can further be combined with various data augmentation methods to obtain better performance.

References

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément

Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Sam Shleifer. 2019. Low resource text classification with ulmfit and backtranslation. *arXiv preprint arXiv:1903.09244*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019a. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer.

Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019b. " mask and infill": Applying masked language model to sentiment transfer. *arXiv preprint arXiv:1908.08039*.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for natural language understanding. *arXiv preprint arXiv:1909.11764*.

Author Index