Insights 2024

The 5th Workshop on Insights from Negative Results in NLP

Proceedings of the Workshop

June 20, 2024

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

Silver

Google

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
317 Sidney Baker St. S
Suite 400 - 134
Kerrville, TX 78028
USA
Tel: +1-855-225-1962
acl@aclweb.org

ISBN 979-8-89176-102-5

Introduction

Publication of negative results is difficult in most fields, and the current focus on benchmark-driven performance improvement exacerbates this situation and implicitly discourages hypothesis-driven research. As a result, the development of NLP models often devolves into a product of tinkering and tweaking, rather than science. Furthermore, it increases the time, effort, and carbon emissions spent on developing and tuning models, as the researchers have little opportunity to learn from what has already been tried and failed.

The mission of the workshop on Insights from Negative Results in NLP is to provide a venue for many kinds of negative results, with the hope that they could yield useful insights and provide a much-needed reality check on the successes of deep learning models in NLP. In particular, we solicit the following types of contributions:

- broadly applicable recommendations for training/fine-tuning, especially if X that didn't work is something that many practitioners would think reasonable to try, and if the demonstration of X's failure is accompanied by some explanation/hypothesis;
- ablation studies of components in previously proposed models, showing that their contributions are different from what was initially reported;
- datasets or probing tasks showing that previous approaches do not generalize to other domains or language phenomena;
- trivial baselines that work suspiciously well for a given task/dataset;
- cross-lingual studies showing that a technique X is only successful for a certain language or language family;
- experiments on (in)stability of the previously published results due to hardware, random initializations, preprocessing pipeline components, etc;
- theoretical arguments and/or proofs for why X should not be expected to work;
- demonstration of issues with under-reporting of training details of pre-trained models, including test data contamination and invalid comparisons.

The fifth iteration of the *Workshop on Insights from Negative Results* attracted 28 submissions and 4 from ACL Rolling Reviews. In terms of topics/themes, 4 papers from our accepted proceedings discussed "zero-shot / few-shot learning / low-resource settings"; 1 discussed "cross-modal fine-tuning"; 6 papers examined pre-trained representations / generalization; 1 dealt with tokenization; 6 on the topic of "LLM Reasoning / Alignment / Evaluations / Probing"; 1 on Multi-task Learning. Some submissions fit in more than one category.

We accepted 19 short papers (57.5% acceptance rate).

We hope the workshop will continue to contribute to the many reality-check discussions on progress in NLP. If we do not talk about things that do not work, it is harder to see what the biggest problems are and where the community effort is the most needed.

Organizing Committee

Organizers

Shabnam Tafreshi, AI inQbator at Evernorth Healthcare & UMD
Arjun Reddy Akula, Google DeepMind, USA
João Sedoc, New York University, USA
Anna Rogers, IT University of Copenhagen, Denmark
Aleksandr Drozd, RIKEN, Japan
Anna Rumshisky, University of Massachusetts Lowell / Amazon Alexa, USA

Program Committee

Chairs

Shabnam Tafreshi, AI inQbator at Evernorth Healthcare and UMD
Arjun Akula, Google DeepMind
João Sedoc, New York University
Anna Rogers, IT University of Copenhagen
Aleksandr Drozd, RIKEN Center for Computational Science
Anna Rumshisky, University of Massachusetts Lowell

Program Committee

Wazir Ali, University of Turku
Nihal Balani, Google
Adrian Benton, Google
Shaun Cassini, University of Sheffield
Chung-Chi Chen, National Institute of Advanced Industrial Science and Technology
Young Min Cho, University of Pennsylvania
Tamás Ficsor, University of Szeged
Salvatore Giorgi, University of Pennsylvania
Edward G o w - S m i t h, University of Sheffield
Kazuma Hashimoto, Google Research
Shreya Havaldar, University of Pennsylvania
Marzena Karpinska, University of Massachusetts Amherst
Neha Nayak Kennard, University of Massachusetts Amherst
Anuj Khare, Google LLC
Huda Khayrallah, Microsoft
Saranya Krishnamoorthy, Evernorth Health Services
Gaurav Kumar, Google
Seolhwa Lee, Technical University of Darmstadt
Yifei Li, University of Pennsylvania
Ashutosh Modi, Indian Institute of Technology Kanpur
Tristan Naumann, Microsoft Research
Juan Navarro Horniacek, Google
John E. Ortega, Northeastern University
Chanjun Park, Upstage
Giovanni Puccetti, Scuola Normale Superiore di Pisa
Jitesh Punjabi, Google LLC
Sunny Rai, University of Pennsylvania
Jordan Rodu, University of Virginia
Ayush Singh, Evernorth Health Services Inc.
Maximilian Spliethöver, Leibniz University Hannover
Mahesh Goud Tandarpally, Amazon
Emil Vatai, Riken R-CCS
Shubham Vatsal, New York University

Table of Contents

Program

Tuesday, June 20, 2023

09:15 - 09:30 *Opening Remarks*

09:30 - 10:30 *Oral Session 1*

An Analysis of BPE Vocabulary Trimming in Neural Machine Translation
Marco Cognetta, Tatsuya Hiraoka, Rico Sennrich, Yuval Pinter and Naoaki Okazaki

Pointer-Generator Networks for Low-Resource Machine Translation: Don't Copy That!
Niyati Bafna, Philipp Koehn and David Yarowsky

On the Limits of Multi-modal Meta-Learning with Auxiliary Task Modulation Using Conditional Batch Normalization
Jordi Armengol - Estape, Vincent Michalski, Ramnath Kumar, Pierre - Luc St-Charles, Doina Precup and Samira Ebrahimi Kahou

WINOVIZ: Probing Visual Properties of Objects Under Different States
Woojeong Jin, Tejas Srinivasan, Jesse Thomason and Xiang Ren

10:30 - 11:00 *Coffee*

11:00 - 11:45 *Invited Talk: Marius Mosbach, Analysis Work in NLP: The Good, the Bad and the Ugly*

11:45 - 12:30 *Oral Session 2*

What explains the success of cross-modal fine-tuning with ORCA?
Paloma Garcia De Herreros, Vagrant Gautam, Philipp Slusallek, Dietrich Klakow and Marius Mosbach

I Have an Attention Bridge to Sell You: Generalization Capabilities of Modular Translation Architectures
Timothee Mickus, Raul Vazquez and Joseph Attieh

Knowledge Distillation vs. Pretraining from Scratch under a Fixed (Computation) Budget
Minh Duc Bui, Fabian Schmidt, Goran Glavaš and Katharina Von Der Wense

12:30 - 14:00 *Lunch*

Tuesday, June 20, 2023 (continued)

14:00 - 14:45 *Oral Session 3*

The Paradox of Preference: A Study on LLM Alignment Algorithms and Data Acquisition Methods
Rishikesh Devanathan, Varun Nathan and Ayush Kumar

Can probing classifiers reveal the learning by contact center large language models?: No, it doesn't!
Varun Nathan, Ayush Kumar and Digvijay Ingle

Multi-Task Learning with Adapters for Plausibility Prediction: Bridging the Gap or Falling into the Trenches?
Annerose Eichel and Sabine Schulte Im Walde

14:45 - 15:30 *Invited Talk: Sasha Luccioni, Reproducibility in ML and the Environment: What's the Connection?*

15:30 - 16:00 *Coffee*

16:00 - 17:00 *Poster Session*

17:00 - 17:10 *Closing Remarks*

MoSECroT: Model Stitching with Static Word Embeddings for Crosslingual Zero-shot Transfer

Haotian Ye*, Yihong Liu*, Chunlan Ma*, and Hinrich Schütze

Center for Information and Language Processing, LMU Munich
Munich Center for Machine Learning (MCML)
{yehao, yihong, chunlan}@cis.lmu.de

Abstract

Transformer-based pre-trained language models (PLMs) have achieved remarkable performance in various natural language processing (NLP) tasks. However, pre-training such models can take considerable resources that are almost only available to high-resource languages. On the contrary, static word embeddings are easier to train in terms of computing resources and the amount of data required. In this paper, we introduce **MoSECroT** (**Mo**del **S**titching with Static Word **E**mbeddings for **Cro**sslingual Zero-shot **T**ransfer), a novel and challenging task that is especially relevant to low-resource languages for which static word embeddings are available. To tackle the task, we present the first framework that leverages relative representations to construct a common space for the embeddings of a source language PLM and the static word embeddings of a target language. In this way, we can train the PLM on source-language training data and perform zero-shot transfer to the target language by simply swapping the embedding layer. However, through extensive experiments on two classification datasets, we show that although our proposed framework is competitive with weak baselines when addressing MoSECroT, it fails to achieve competitive results compared with some strong baselines. In this paper, we attempt to explain this negative result and provide several thoughts on possible improvement.

1 Introduction

The emergence of PLMs and their multilingual counterparts (mPLMs) (Devlin et al., 2019; Conneau et al., 2020) have proven effective for various NLP tasks (Artetxe et al., 2020; ImaniGooghari et al., 2023). However, such models are mostly limited to no more than a hundred languages, as the pre-training requires considerable data that is only available to these languages, leaving the majority of the world's low-resource languages uncovered. In this work, we explore the possibility of leveraging (1) a PLM in a source language, (2) static word embeddings in a target language, which are readily available for many low-resource languages and are much easier to train, and (3) a technique called model stitching, to enable zero-shot on the target language without the need to pre-train.

Our contribution is summarized as follows: (i) we introduce **MoSECroT**, a novel and challenging task for (especially low-resource) languages where static word embeddings are available. (ii) We propose a solution that leverages relative representations to construct a common space for source (English in our case) and target languages and that allows zero-shot transfer for the target languages.

2 Related Work

Aligned crosslingual word embeddings enable transfer learning by benefiting from a shared representation space for the source and target languages. Such embedding pairs are typically either trained jointly (Hermann and Blunsom, 2014; Vulic and Moens, 2016) or obtained through post-alignment (Lample et al., 2018; Artetxe et al., 2018). Our work applies a transformation in the manner of the latter to align two embedding spaces where the source embeddings are derived from a PLM and target embeddings are static word embeddings.

Based on a recent consensus that similar inner representations are learned by neural networks regardless of their architecture or domain (Kornblith et al., 2019; Vulić et al., 2020), Moschella et al. (2023) propose an approach to align latent spaces with respect to a set of samples, called parallel anchors. They transform the original, absolute space to one defined by relative coordinates of the parallel anchors, and denote all the transformed samples in the relative coordinates as relative representations.

Model stitching was proposed as a way to com-

*Equal contribution.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 1–7
June 20, 2024 ©2024 Association for Computational Linguistics

bine (stitch together) components of different neural models. Trainable stitching layers are first introduced by Lenc and Vedaldi (2015), with a series of subsequent works demonstrating the effectiveness of the approach (Bianchi et al., 2020; Bansal et al., 2021).

3 MoSECroT Task Setting

The task setting is straightforward: given a PLM of a high-resource language (regarded as the source language) and static word embeddings of another language (low-resource and regarded as the target language), the goal is to achieve zero-shot transfer by using the target language embeddings directly with the source language model via embedding layer stitching. This can be done by first applying an alignment between the source and target embedding spaces and subsequently swapping the embedding matrices of the PLM.

We propose a novel method that leverages relative representations for embedding space mapping. In the following, we describe our methodology in more detail.

4 Methodology

Parallel anchor selection We first extract bilingual parallel lexica between the source and the target language. For most high-resource languages, large bilingual lexica are available from MUSE[1]. For low-resource languages, we crawl translations of source language vocabulary from PanLex[2] and Google Translate[3]. Then we derive a subset of the lexica as the parallel anchors A for our method: we only keep those parallel lexica which exist in the embeddings of source and target languages[4].

Relative representations Following Moschella et al. (2023), we build relative representations (RRs) for each token in the embedding space based on their similarities with anchor tokens in the respective language. Specifically, we compute the cosine similarity of the embedding of each token with the embedding of each anchor token. This computation is done in the embedding spaces of the source and target languages respectively. For example, in the source language, the similarity between token x_i and anchor a_j is calculated as follows:

$$r^s_{(i,j)} = \text{cos-sim}(\boldsymbol{E}^s_{\{x_i\}}, \boldsymbol{E}^s_{\{a_j\}})$$

where $\boldsymbol{E}^s_{\{x_i\}}$, $\boldsymbol{E}^s_{\{a_j\}}$ are the word embedding of x_i and a_j in the source PLM embeddings $\boldsymbol{E}^s$. The relative representation of token x_i from the source language is then defined as follows:

$$\boldsymbol{R}^s_{\{x_i\}} = [r^s_{(i,1)}, r^s_{(i,2)}, r^s_{(i,3)}, \cdots, r^s_{(i,|A|)}]$$

Note that the relative representation is sensitive to the order of the anchors, so the relative representation for each token is computed with the anchors in the same order. This computation results in a matrix $\boldsymbol{R}^s \in \mathbb{R}^{|V^s| \times |A|}$ of source language embeddings and a matrix $\boldsymbol{R}^t \in \mathbb{R}^{|V^t| \times |A|}$ of target language embeddings, where $|V^s|$ (resp. $|V^t|$) is the source-language (resp. target-language) vocabulary size and $|A|$ is the number of parallel anchors.

Embedding mapping The obtained relative representations are vectors in $\mathbb{R}^{|A|}$ for both source and target languages. This dimension does not suit the hidden dimension of the Transformer body of the source PLM. Therefore, we propose to map the relative representations of both source and target languages back to $\mathbb{R}^D$, which is the same as the dimension of $\boldsymbol{E}^s$. Given $\boldsymbol{E}^s$ and $\boldsymbol{R}^s$ for source language (resp. $\boldsymbol{E}^t$ and $\boldsymbol{R}^t$ for target language), we compute the transformed embedding of any token x_i from the source language (resp. any token y_i from the target language) as follows:

$$\boldsymbol{F}^s_{\{x_i\}} = \frac{\sum_{n \in \mathbb{N}(x_i)} (\boldsymbol{R}^s_{\{x_i\},n} / \tau \cdot \boldsymbol{E}^s_{\{n\}})}{\sum_{n \in \mathbb{N}(x_i)} \boldsymbol{R}^s_{\{x_i\},n} / \tau}$$

$$\boldsymbol{F}^t_{\{y_i\}} = \frac{\sum_{n \in \mathbb{N}(y_i)} (\boldsymbol{R}^t_{\{y_i\},n} / \tau \cdot \boldsymbol{E}^s_{\{n\}})}{\sum_{n \in \mathbb{N}(y_i)} \boldsymbol{R}^t_{\{y_i\},n} / \tau}$$

where $\mathbb{N}(x_i)$ (resp. $\mathbb{N}(y_i)$) is the set of top-k closest anchors in terms of the cosine similarity recorded in $\boldsymbol{R}^s_{x_i}$ (resp. $\boldsymbol{R}^t_{y_i}$), $\boldsymbol{R}^s_{\{x_i\},n}$ (resp. $\boldsymbol{R}^s_{\{y_i\},n}$) is the cosine similarity between $\boldsymbol{E}^s_{\{x_i\}}$ (resp. $\boldsymbol{E}^t_{\{y_i\}}$) and $\boldsymbol{E}^s_{\{n\}}$ (resp. $\boldsymbol{E}^t_{\{n\}}$), and τ is the temperature. Note that both the resulting transformed embeddings $\boldsymbol{F}^s_{\{x_i\}}$ and $\boldsymbol{F}^t_{\{y_i\}}$ are in $\mathbb{R}^D$, because it is a weighted sum of the anchor embedding in the **source language**, i.e., $\boldsymbol{E}^s_{\{n\}}$. A simple summary of the process is to *represent any token, no matter whether it is from the source or target language, as a weighted sum of the embeddings of some parallel anchors in the source-language embedding space.*

[1] https://github.com/facebookresearch/MUSE
[2] https://panlex.org
[3] https://translate.google.com
[4] The source language is always English and its embeddings are extracted from English BERT's (Devlin et al., 2019) token embeddings. For target languages, embeddings are static word embeddings from fastText (Bojanowski et al., 2017).

Zero-shot stitching So far we project the target-language embeddings to $\mathbb{R}^D$, which suits the hidden dimension of the Transformer body of the source language. We also manipulate the original token embedding matrix of the source language, where the matrix dimensions stay the same: $\boldsymbol{F}^s \in \mathbb{R}^{|V^s| \times D}$. We can simply fine-tune the model ($\boldsymbol{F}^s$ and the Transformer body) on the source-language train set of a downstream task and then assemble a target-language model for zero-shot transfer, without training on the target language. To do this, we only need to swap the source-language embeddings $\boldsymbol{F}^s$ with target-language embeddings $\boldsymbol{F}^t$.

5 Experiments

5.1 Setup

We use the cased version of the English BERT model (`bert-base-cased`) as the source language PLM and consider eight target languages. Three of the target languages are high-resource: German (**de**), Spanish (**es**), and Chinese (**zh**), and the rest are low-resource: Faroese (**fo**), Maltese (**mt**), Eastern Low German (**nds**), Sakha (**sah**), and Tatar (**tt**). Pre-trained static embeddings for all target languages are available from fastText[5], except for Eastern Low German, for which we download fastText embeddings from Huggingface[6].

Using the method proposed in §4, we obtain pairwise parallel anchors between English and each target language. The size of the anchor set varies depending on the vocabulary size of the language's embeddings and the overlap between the English and target language lexica, which is the following for each target language: 11836 (en-de), 11395 (en-es), 7662 (en-zh), 1577 (en-fo), 2600 (en-mt), 1309 (nds), 3242 (en-sah), and 9275 (en-tt).

We evaluate the proposed method on two text classification datasets: Multilingual Amazon Reviews Corpus (Keung et al., 2020) and Taxi1500 (Ma et al., 2023). See §C for details.

Apart from the standard weighting scheme illustrated in §4, we propose two more settings: one where we apply softmax over relative representation weights (in the **Embedding mapping** step), and another using sparsemax (Martins and Astudillo, 2016). Compared to softmax, sparsemax produces sparse weight distributions, meaning more similarities are concentrated on fewer anchors. We conduct preliminary experiments to identify the optimal top-k closest anchors $\in \{1, 10, 50, 100\}$ and find that the results are best when using the top 50 anchors. See §A for an exploration of how different choices of k influence the performance.

	de	es	zh
LR	0.52	0.51	0.50
mBERT	**0.61**	**0.65**	**0.51**
LS	0.46	0.46	0.30
RRs standard top-50	0.53	0.51	0.38
RRs softmax top-50	0.50	0.53	0.38
RRs sparsemax top-50	0.56	0.57	0.24

Table 1: Evaluation results on the Amazon Reviews Corpus. We report macro F_1 scores on the test sets of three high-resource target languages. **Bold**: highest score per column.

5.2 Baselines

We compare our method against three baselines:

Logistic Regression (LR) We train a simple target language logistic regression classifier using the average of static word embeddings of the input sentences. This approach does not require expensive training of a language model but assumes we have sufficient target language training data for a specific downstream task, which is hardly the case for most low-resource languages in real scenarios.

mBERT We fine-tune multilingual BERT (mBERT) (Devlin et al., 2019), which is pre-trained on more than 100 languages, using the English training data, and perform zero-shot predictions directly on the target language test data.

Least squares projection (LS) We propose a straightforward approach, inspired by embedding alignment frameworks such as VecMap (Artetxe et al., 2018), to project target language embeddings into the same space as the English PLM embeddings. Specifically, we learn a transformation matrix $\boldsymbol{W} \in \mathbb{R}^{D^t \times D}$ by minimizing $||\boldsymbol{A}^t\boldsymbol{W} - \boldsymbol{A}^s||_F^2$, where $\boldsymbol{A}^t \in \mathbb{R}^{|A| \times D^t}$ is the embeddings of anchors in the target language and $\boldsymbol{A}^s \in \mathbb{R}^{|A| \times D}$ is the embeddings of anchors from the English PLM. We then project all target language embeddings using

[5] `https://fasttext.cc/docs/en/pretrained-vectors.html`

[6] `https://huggingface.co/facebook/fasttext-nds-vectors`

	de	es	zh	mt	sah	fo	nds	tt
LR	**0.30**	0.32	0.56	**0.38**	**0.48**	**0.47**	**0.18**	**0.43**
mBERT	0.24	**0.60**	**0.62**	0.08	0.07	0.18	0.12	0.18
LS	0.14	0.26	0.24	0.08	0.12	0.06	0.08	0.07
RRs standard top50	0.20	0.44	0.28	0.14	0.16	0.16	0.06	0.14
RRs softmax top50	0.20	0.48	0.28	0.15	0.19	0.16	0.06	0.17
RRs sparsemax top50	0.24	0.37	0.13	0.15	0.18	0.20	0.13	0.21

Table 2: Evaluation results on the Taxi1500 dataset. Reported metrics are macro F_1 scores on the test sets of eight target languages. Scores are averaged over five runs with different random seeds. **Bold**: highest score per column.

$\boldsymbol{W}$ and replace the BERT embedding layer with the resulting matrix.

5.3 Results

We present evaluation results of RRs with the proposed settings (§5.1) and compare them with the baselines in Tables 1 and 2. Macro F_1 is used due to class imbalance in both datasets.

We notice that the naive LS baseline is almost always beaten by the proposed method under multiple RR settings on both datasets. The only exception is nds, in Table 2, where both LS and RRs perform badly. This observation is a strong indicator that RRs can better leverage the semantic similarity encoded in different types of embeddings than LS.

Not very surprisingly, zero-shot with mBERT is effective for high-resource languages in both datasets but underperforms LR with large gaps on low-resource languages in Taxi1500. There are two possible explanations for this phenomenon. First, representations in mBERT are not well-aligned across low-resource languages. This is possibly due to data sparsity, which is observed by previous work (Wu and Dredze, 2020), where mBERT archives good performance on high-resource languages but sub-optimal performance on low-resource languages. Second, Taxi1500 is a relatively easy task: a model with good alignment across languages, especially on the word level, is expected to perform well. This argument is supported by a previous work (Liu et al., 2023), where well-aligned word embeddings achieve better zero-shot crosslingual performance than mPLMs on a wide range of languages in Taxi1500.

Although none of the RR settings outperforms mBERT on high-resource languages (as mentioned earlier, mBERT has strong crosslingual transfer ability on high-resource languages), for all five low-resource languages not seen by mBERT (mt, sah, fo, nds, tt), RRs outperform mBERT consistently, with varying margins (ranging from +0.12 for sah to +0.01 for nds). This suggests that RRs can be a promising alternative when a low-resource language is not covered by an mPLM.

6 Analysis

In this section, we want to propose possible reasons for the suboptimal results obtained by our framework tackling the MoSECroT task.

Anchor selection The quality of the parallel anchors largely relies on the quality of the bilingual lexica, which may contain, among others, polysemous words, that may influence the alignment quality. Normalization can also be a source of ambiguity. For example, MUSE converts all words into lowercase, so the word `sie` can have three meanings in the German-English lexicon: `you`, `she`, and `they`. We (1) only consider one translation (if there are multiple) for each target language word, which may not be the most accurate one; and (2) treat all target language words whose translations are in the source language vocabulary as anchors, which increases the frequency of noisy translation pairs.

We try to decrease the influence of potentially noisy anchor pairs by reducing the number of anchors to 3000 and 500 (the original anchor set used during the preliminary experiments contains 6731 anchors, see §4) through random sampling, following the observation by Moschella et al. (2023) that uniform selection from an anchor set is both straightforward and has good performance. We also remove stop words, whose translations are more unstable, from the anchor set. Neither of the two modifications shows an improvement over the full anchor set (see §B for the comparison). One possible explanation is that the translation qualities vary across anchors and thus we cannot predict the quality of sampled anchors.

Translation quality We find that a large portion of translations retrieved from PanLex are of low quality. This is partly due to PanLex using intermediate languages when direct translation is unavailable for the language pair. We filter the translations by empirically setting a threshold to the translation quality scores, available through the API for every translation. Nevertheless, we note that a high translation quality score does not guarantee the translation is perfect, and many translations are good despite having low translation quality scores. We believe the lack of high-quality parallel lexica is a possible reason that RRs do not reach their full potential on low-resource languages.

Reinitialized embedding space Our method requires swapping the original PLM embeddings with the transformed English RRs before fine-tuning on English data, whereas the embedding space of RRs might diverge substantially from the original embedding space. As a result, it is unclear whether the rest of the model parameters can be adapted to the new embeddings during fine-tuning, especially on smaller datasets like Taxi1500. We thus suggest the alteration of the embedding space through reinitialization with RRs as a likely factor as to why we do not achieve good performance.

7 Conclusion

In this work, we introduce MoSECroT, a novel and challenging task that is relevant for, in particular, low-resource languages for which static word embeddings are available but few resources exist. In addition, we propose for the first time a method that leverages relative representations for embedding space mapping and enables zero-shot transfer. Specifically, we fine-tune a monolingual English language model using only English data, swap the embeddings with target language embeddings aligned using RRs, and apply zero-shot evaluation on the target language. We show that the proposed method is promising compared with mBERT on unseen languages but only modest improvements are achieved. We provide several possible reasons and leave improvement possibilities for future research.

Limitations

In this work, we propose the task of MoSECroT and a solution to leverage available static pretrained embeddings and tackle downstream tasks for low-resource languages. Our work has a few limitations open to future research. First, we only experiment with one model architecture (BERT). Although many language-specific BERT models exist and thus our method is applicable to a wide range of high-resource source languages, it would nevertheless be interesting to compare performance across different model architectures. Second, the explored tasks are exclusively text classification tasks. We expect that the robustness of our method can be much better studied by applying it to a more diverse set of tasks.

Acknowledgements

This work was funded by the European Research Council (grant #740516).

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

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*, pages 4623–4637, Online. Association for Computational Linguistics.

Yamini Bansal, Preetum Nakkiran, and Boaz Barak. 2021. Revisiting model stitching to compare neural representations. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 225–236.

Federico Bianchi, Jacopo Tagliabue, Bingqing Yu, Luca Bigon, and Ciro Greco. 2020. Fantastic embeddings and how to align them: Zero-shot inference in a multi-shop scenario. In *Proceedings of the SIGIR 2020 eCom workshop, July 2020, Virtual Event, published at http://ceur-ws.org (to appear)*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

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.

Karl Moritz Hermann and Phil Blunsom. 2014. Multilingual models for compositional distributed semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 58–68, Baltimore, Maryland. Association for Computational Linguistics.

Ayyoob ImaniGooghari, Peiqin Lin, Amir Hossein Kargaran, Silvia Severini, Masoud Jalili Sabet, Nora Kassner, Chunlan Ma, Helmut Schmid, André Martins, François Yvon, and Hinrich Schütze. 2023. Glot500: Scaling multilingual corpora and language models to 500 languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1082–1117, Toronto, Canada. Association for Computational Linguistics.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. 2020. The multilingual Amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4563–4568, Online. Association for Computational Linguistics.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3519–3529. PMLR.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Karel Lenc and Andrea Vedaldi. 2015. Understanding image representations by measuring their equivariance and equivalence. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 991–999. IEEE Computer Society.

Yihong Liu, Haotian Ye, Leonie Weissweiler, Renhao Pei, and Hinrich Schuetze. 2023. Crosslingual transfer learning for low-resource languages based on multilingual colexification graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8376–8401, Singapore. Association for Computational Linguistics.

Chunlan Ma, Ayyoob ImaniGooghari, Haotian Ye, Ehsaneddin Asgari, and Hinrich Schütze. 2023. Taxi1500: A multilingual dataset for text classification in 1500 languages.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1614–1623, New York, New York, USA. PMLR.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. 2023. Relative representations enable zero-shot latent space communication. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Ivan Vulic and Marie-Francine Moens. 2016. Bilingual distributed word representations from document-aligned comparable data. *J. Artif. Intell. Res.*, 55:953–994.

Ivan Vulić, Sebastian Ruder, and Anders Søgaard. 2020. Are all good word vector spaces isomorphic? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3178–3192, Online. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual BERT? In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120–130, Online. Association for Computational Linguistics.

A Number of closest anchors

In addition to using all (6731) parallel anchors, we consider only the top-k ($k \in \{1, 10, 50, 100\}$) closest anchors of each word. We identify the optimal value for k closest anchors based on zero-shot performance on German and Chinese portions of the Amazon Reviews Corpus (§C.1). Table 3 shows results for different k values.

k	de	zh
1	0.44	0.41
10	0.51	0.38
50	0.50	0.40
100	0.51	0.38
6731	0.44	0.21

Table 3: Number of closest parallel anchors (k) and the corresponding zero-shot performance on de and zh portions of the Amazon Reviews Corpus.

B Total number of anchors

Following Moschella et al. (2023), we randomly sample a subset of the parallel anchors ($|\boldsymbol{A}| \in \{500, 3000\}$), and exclude stop words from the anchor set. Table 4 shows zero-shot performance on German and Chinese portions of the Amazon Reviews Corpus (§C.1).

$\|\boldsymbol{A}\|$	de	zh
500	0.39	0.19
3000	0.19	0.19
6731	0.44	0.21

Table 4: The total number of parallel anchors and the corresponding zero-shot performance on de and zh portions of the Amazon Reviews Corpus.

C Evaluation datasets

C.1 Multilingual Amazon Reviews Corpus

Presented by Keung et al. (2020) and containing product reviews in six languages, the original dataset uses five labels corresponding to star ratings, which we aggregate into three classes: positive, neutral, and negative. We evaluate the three high-resource target languages (de, es, zh) on this dataset.

C.2 Taxi1500

Taxi1500 (Ma et al., 2023) is a classification dataset containing six classes for more than 1500 languages, including all of our target languages. We follow the authors' original training procedure and hyperparameters and use a learning rate of 1e-5 instead of 2e-5, which we find works better for our settings.

D Computational resources

Training can be completed in under three hours on eight NVIDIA GeForce GTX 1080 Ti GPUs for the Multilingual Amazon Reviews Corpus or about half an hour on a single NVIDIA GeForce GTX 1080 Ti GPU for Taxi1500.

What explains the success of cross-modal fine-tuning with ORCA?

Paloma García-de-Herreros[*1] **Vagrant Gautam**[*1] **Philipp Slusallek**[1, 2]
Dietrich Klakow[1] **Marius Mosbach**[3,4]
[1]Saarland University [2]DFKI [3]McGill University [4]Mila – Quebec AI Institute
{pgherreros,vgautam}@lsv.uni-saarland.de

Abstract

ORCA (Shen et al., 2023) is a recent technique for cross-modal fine-tuning, i.e., applying pre-trained transformer models to modalities beyond their training data. The technique consists primarily of training an embedder and fine-tuning the embedder and model. Despite its high performance on a variety of downstream tasks, we do not understand precisely how each of these components contribute to ORCA's success. Therefore, we run a series of ablations and find that embedder training does not help 2D tasks at all, contrary to what the original paper posits. In 1D tasks, some amount of embedder training is necessary but more is not better. In 4 out of 6 datasets we experiment with, it is model fine-tuning that makes the biggest difference. Through our ablations and baselines, we contribute a better understanding of the individual components of ORCA.

1 Introduction

Modern AI is based on a pipeline of pre-training general-purpose models on vast amounts of data and then adapting them to specific tasks. Examples across natural language processing (NLP) and computer vision (CV) typically focus on within-modality adaptation across, e.g., tasks or domains, but there is also a recent line of work that looks at leveraging pre-trained models *across* modalities, e.g., Frozen Pretrained Transformers (FPT) (Lu et al., 2021), ORCA (Shen et al., 2023), OmniPred (Song et al., 2024), Unified PDE Solver (UPS) (Shen et al., 2024), *inter alia*.

ORCA is a recent example of a method for cross-modal fine-tuning (Shen et al., 2023). It consists of a three-phase pipeline, shown in Figure 1. First, a pre-trained transformer model is chosen, and a custom embedder and predictor are created to support new tasks with any input and output dimensions. Second, a within-modality proxy dataset is chosen. The embedder is trained to minimize the distance between the target dataset and this proxy dataset, in order to map the target dataset into the embedding space of the model. Finally, all three components are fine-tuned on data from the target task.

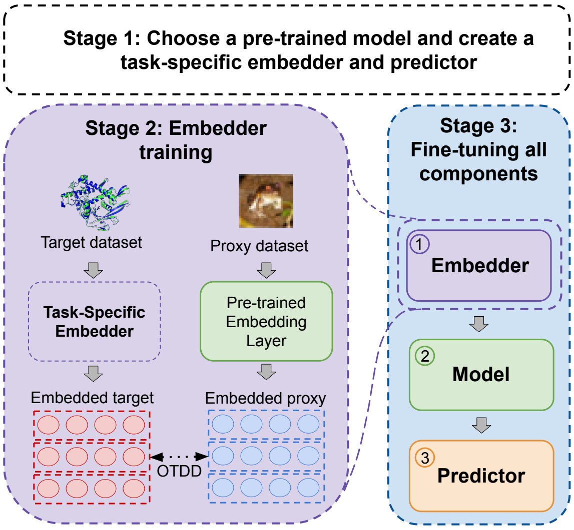


Figure 1: The ORCA pipeline. Stage 2 involves training the task-specific embedder. Stage 3 fine-tunes the embedder, the pre-trained encoder, and the predictor.

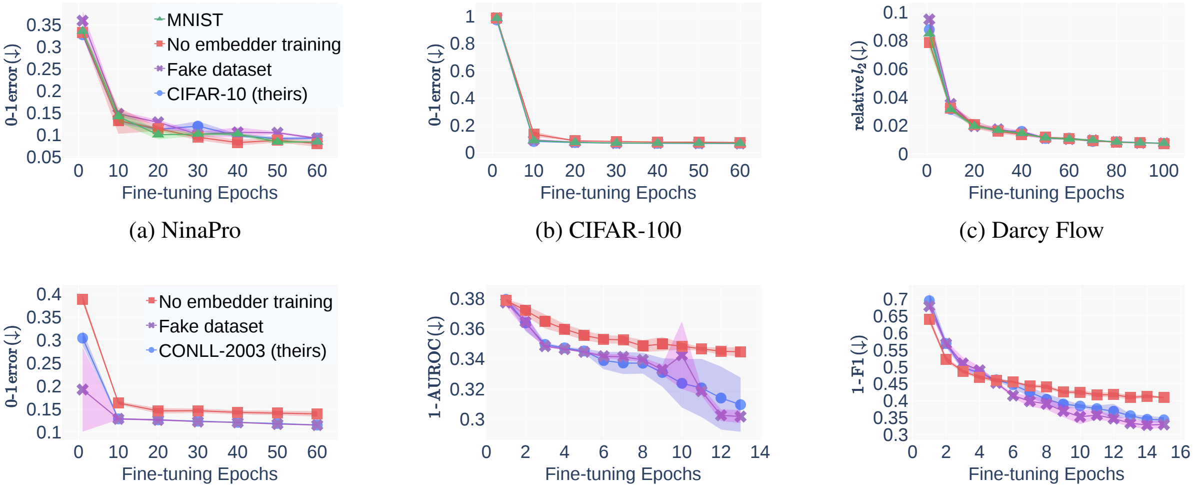
According to Shen et al. (2023), embedder training is the reason for ORCA's success. We expand on their ablations to better understand the contributions of ORCA's individual components, focusing on ablating the second and third stages of the pipeline. Our specific research questions are:

1. How does the choice of proxy dataset affect performance? (§3)
2. Does doing (more) embedder training improve performance? (§4)
3. What do the embedder and the pre-trained model contribute individually? (§5)
4. How much pre-training is necessary for cross-modal transfer? (§6)

By disentangling the contributions of embedder training and model fine-tuning, our results provide a more nuanced perspective on the success of cross-modal fine-tuning with ORCA. Additionally, our findings highlight the importance of strong baselines and careful ablations when making claims about *why* a method works.

[*] Equal contribution.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 8–16
June 20, 2024



(a) NinaPro (b) CIFAR-100 (c) Darcy Flow

(d) Satellite (e) DeepSEA (f) ECG

Figure 2: Per-epoch fine-tuning performance (↓) on 2D tasks (above) and 1D tasks (below) when the embedder is trained with different proxy datasets or not trained at all, i.e., naive fine-tuning.

2 Experimental setup

Unless otherwise specified, we follow the ORCA paper in using RoBERTa-base (Liu et al., 2019) and Swin-base (Liu et al., 2021) as the pre-trained transformers, a convolutional architecture for the embedder, and a linear transformation for the predictor (see Appendix C for details). We also use optimal transport dataset distance (OTDD; Alvarez-Melis and Fusi, 2020) as the loss function during embedder training. All our experiments use their publicly available code.[1] For training, we use the same hyperparameters as they do, except for the batch size when training on Satellite (64) and ECG (32) data. We evaluate on six target datasets that appear in the original paper, chosen to represent all pairs of dimensions and types, and we experiment with various proxy datasets. Dataset details are shown in Appendix B.

Target datasets. We select three 2D datasets (NinaPro, CIFAR-100, and Darcy Flow) and three 1D datasets (Satellite, DeepSEA, and ECG) from the NAS-Bench-360 benchmark (Tu et al., 2022). 2D and 1D refer to the input being either a matrix (2 dimensions) or a sequence (1 dimension).

Proxy datasets. The original paper uses CIFAR-10 (Krizhevsky, 2009) as the proxy dataset for all 2D tasks, and CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003) for all 1D tasks. We experiment with additional proxy datasets to analyze their influence on overall performance.

For the 2D tasks, we compare to two other image datasets that maintain the same number of classes: MNIST (Deng, 2012), a different image dataset, and Fakedata[2], a dataset of randomly classified white noise images (Paszke et al., 2019). For the 1D tasks, we compare to a custom-created fake dataset classifying randomly generated language feature vectors into the same number of classes as CoNLL.

3 How does the choice of proxy dataset affect performance?

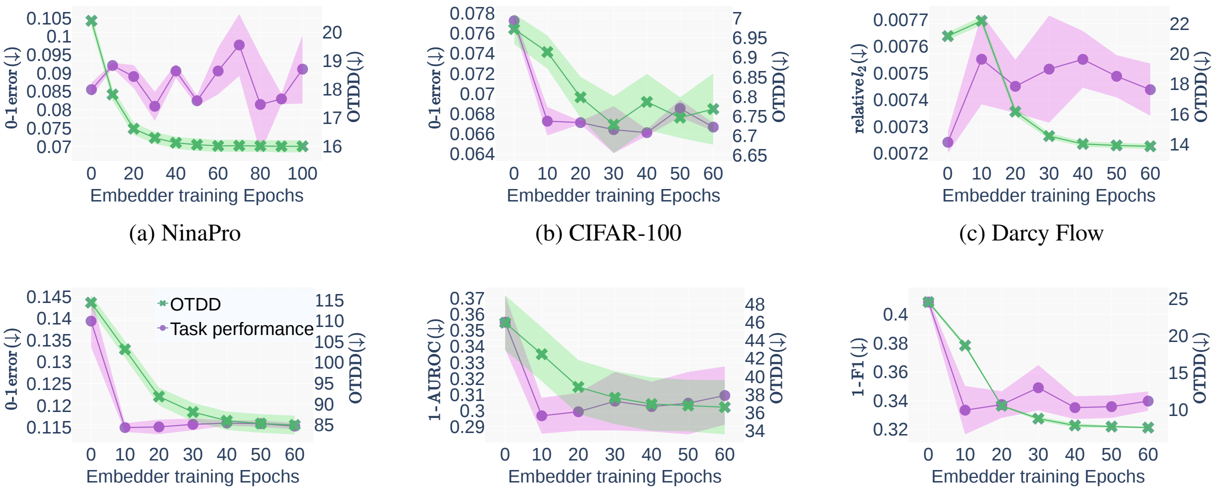
In this section, we experiment with the choice of proxy dataset for the tasks. As a baseline, we compare to just fine-tuning the embedder, model and predictor, without training the embedder first.

As Figure 2 shows, all fine-tuning curves for the 2D datasets (first row) overlap, indicating that the choice of proxy dataset is not important. Even fake data as a proxy dataset results in the same performance. Similarly, for the 1D tasks (second row), there is no real difference between using CoNLL and fake embeddings. Together, this shows that **the choice of proxy dataset for embedder training does not matter for ORCA to work**.

Comparing to a naive fine-tuning baseline allows us to evaluate the claim that "ORCA consistently outperforms naive fine-tuning" (Shen et al., 2023). We find that **embedder training does play a role in the 1D tasks, but does not matter for 2D tasks**, even in the early stages of fine-tuning.

[1] https://github.com/sjunhongshen/ORCA/

[2] From torchvision.datasets.



(a) NinaPro (b) CIFAR-100 (c) Darcy Flow

(d) Satellite (e) DeepSEA (f) ECG

Figure 3: Per-epoch embedder training comparing OTDD (↓) (metric minimized during this stage) to downstream task performance (↓).

4 (More) embedder training is not the secret to ORCA's success

The previous results motivate us to more closely examine the role of embedder training in ORCA. In this stage, the OTDD metric is used to quantify the distance between the proxy and target embeddings. The authors minimize OTDD, claiming that "as the dataset distance decreases, the fine-tuning accuracy increases" (Shen et al., 2023).

However, when we examine the relationship between OTDD and downstream task performance, we find that **embedder training is unnecessary in two out of six tasks** (Figures 3a and 3c). For the remaining four tasks, **training the embedder more can even lead to worse task performance**.

As this section and the previous one show that embedder training does not affect final performance on the 2D tasks, we focus on the 1D tasks for our remaining experiments.

5 Which components of ORCA are really necessary?

To better understand how the fine-tuning phase affects the multiple components of ORCA, we experiment with freezing different parts of the pipeline: the embedder, the pre-trained model, or both. We compare our results with the original setup.

Row 1 of Figure 4 shows the results of freezing both the embedder and the pre-trained model, and only fine-tuning the predictor. Across all datasets, the frozen versions perform much worse than the original setup, regardless of embedder training. This indicates that these datasets are not simple enough to be solved by training a simple predictor.

In row 2, we freeze only the pre-trained model, but fine-tune the embedder and the predictor. These frozen versions also perform much worse than the original setup, indicating that **fine-tuning the pre-trained model is a critical component of ORCA**, regardless of dataset and embedder training.

Finally, in row 3, we only freeze the embedder, allowing the fine-tuning stage to affect both the model and the predictor. As we already saw in Figure 2, training the embedder is important across all three datasets. However, once this training is done, even if it is frozen, adapting the pre-trained model is sufficient for good task performance. This shows that **while training the embedder is important for ORCA's success on these datasets, it need not be fine-tuned beyond that**.

6 Pre-training is not always necessary

Our previous results show that fine-tuning the model is necessary for good downstream task performance, but they do not show whether using *pre-trained* models is necessary for this. To answer this question, we use RoBERTa models pre-trained on different amounts of English data. Specifically, we compare the original RoBERTa-base model to a randomly initialized model with no training data, along with three variants trained on less data (Warstadt et al., 2020), shown in Appendix F.

Figure 5 shows that performance varies widely

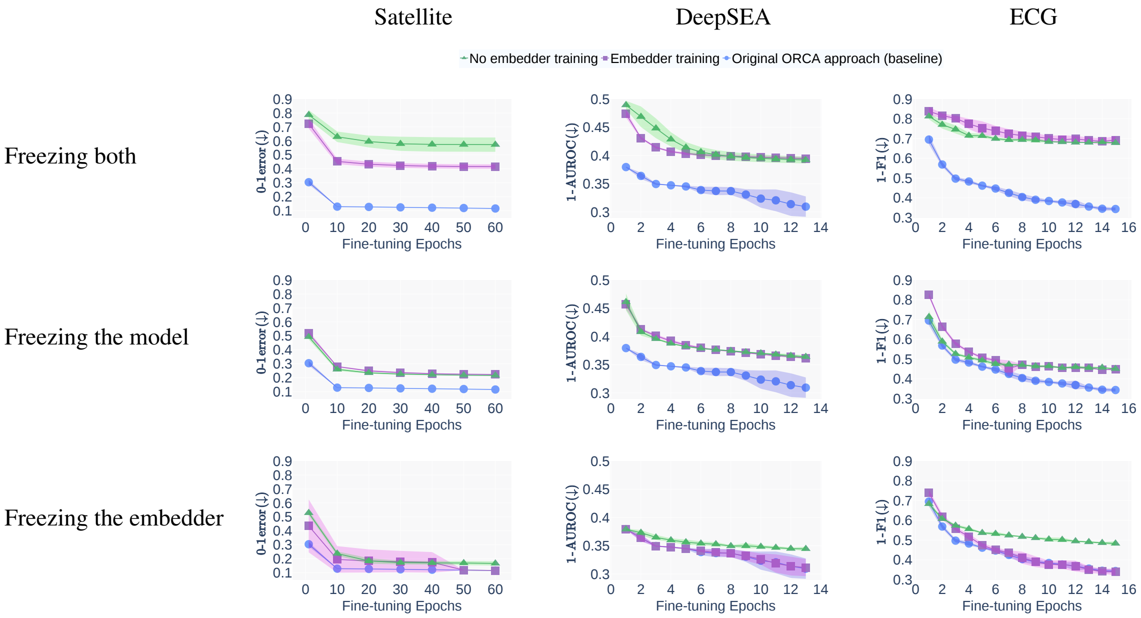
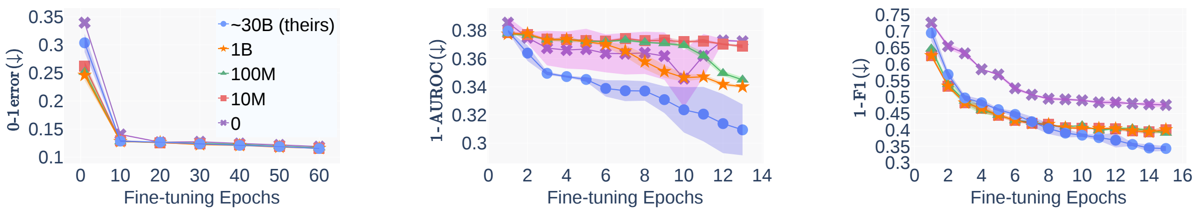


Figure 4: Performance (↓) when freezing both the embedder and model (top row), just the model (middle row) or just the embedder (bottom row), before full fine-tuning. We also evaluate the impact of training (purple squares) vs. not training (green triangles) the embedder before freezing. All ablations are compared to the original ORCA approach without freezing (blue circles).



(a) Satellite (b) DeepSEA (c) ECG

Figure 5: Effect of different amounts of pre-training data on downstream performance (↓).

depending on the dataset. For Satellite, all models perform the same, showing that the task is simple enough to be solved even without pre-training. With DeepSEA and ECG, on the other hand, pre-training data on the scale of 30B tokens results in clearly better performance. These results highlight the importance of comparing to a no pre-training baseline, for ORCA—and indeed all cross-modal fine-tuning work—to ensure that pre-training is actually necessary for the success of the method.

Until the 30B data scale, however, DeepSEA performance remains within the variance of simply fine-tuning a randomly-initialized model, whereas ECG does benefit from even a small amount of pre-training. This shows that even for non-trivial tasks, **the amount of pre-training has a noticeable effect only at certain scales**.

7 Conclusion

We perform a series of ablations to investigate how the different components of ORCA, a recently-proposed method for cross-modal fine-tuning, affect its performance. Contrary to the original results, we find that embedder training does not help 2D tasks at all, compared to just fine-tuning without training the embeddder. In 1D tasks, some amount of embedder training is necessary, but unlike the claim in the original paper, more embedder training can even hurt performance on the target task. When we freeze various components of the ORCA pipeline, we find that fine-tuning the model is crucial for good task performance. Finally, we find that for one of the 1D tasks, using a pre-trained model is actually not necessary, indicating the importance of no pre-training baselines in evaluations of cross-modal transfer.

References

Krizhevsky Alex. 2009. Learning multiple layers of features from tiny images. *https://www. cs. toronto. edu/kriz/learning-features-2009-TR. pdf*.

David Alvarez-Melis and Nicolo Fusi. 2020. Geometric dataset distances via optimal transport. In *Advances in Neural Information Processing Systems*, volume 33, pages 21428–21439. Curran Associates, Inc.

Manfredo Atzori, Arjan Gijsberts, Simone Heynen, Anne-Gabrielle Mittaz Hager, Olivier Deriaz, Patrick Van Der Smagt, Claudio Castellini, Barbara Caputo, and Henning Müller. 2012. Building the ninapro database: A resource for the biorobotics community. In *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pages 1258–1265. IEEE.

Gari D Clifford, Chengyu Liu, Benjamin Moody, H Lehman Li-wei, Ikaro Silva, Qiao Li, AE Johnson, and Roger G Mark. 2017. Af classification from a short single lead ecg recording: The physionet/computing in cardiology challenge 2017. In *2017 Computing in Cardiology (CinC)*, pages 1–4. IEEE.

Li Deng. 2012. The MNIST database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142.

EA Feingold, PJ Good, MS Guyer, S Kamholz, L Liefer, K Wetterstrand, FS Collins, TR Gingeras, D Kampa, EA Sekinger, et al. 2004. The encode (encyclopedia of dna elements) project. *Science*, 306(5696):636–640.

Alex Krizhevsky. 2009. Learning multiple layers of features from tiny images. Technical report.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. 2020. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*.

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.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin Transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022.

Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. 2021. Pretrained transformers as universal computation engines. *arXiv preprint*.

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 32*, pages 8024–8035. Curran Associates, Inc.

François Petitjean, Jordi Inglada, and Pierre Gancarski. 2012. Satellite image time series analysis under time warping. *IEEE Transactions on Geoscience and Remote Sensing*, 50(8):3081–3095.

Junhong Shen, Liam Li, Lucio M. Dery, Corey Staten, Mikhail Khodak, Graham Neubig, and Ameet Talwalkar. 2023. Cross-modal fine-tuning: Align then refine. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 31030–31056. PMLR.

Junhong Shen, Tanya Marwah, and Ameet Talwalkar. 2024. Ups: Towards foundation models for pde solving via cross-modal adaptation. *arXiv preprint arXiv:2403.07187*.

Shu Shen, Kang Gu, Xin-Rong Chen, Ming Yang, and Ru-Chuan Wang. 2019. Movements classification of multi-channel semg based on cnn and stacking ensemble learning. *IEEE Access*, 7:137489–137500.

Xingyou Song, Oscar Li, Chansoo Lee, Daiyi Peng, Sagi Perel, Yutian Chen, et al. 2024. OmniPred: Language models as universal regressors. *arXiv preprint*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Renbo Tu, Nicholas Roberts, Mikhail Khodak, Junhong Shen, Frederic Sala, and Ameet Talwalkar. 2022. NAS-bench-360: Benchmarking neural architecture search on diverse tasks. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

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.

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.

A Limitations

Choice of datasets. We only experiment with three 2D datasets and three 1D datasets, and we do not consider the experiments from the original paper on tabular data, where our findings may not hold. Additionally, due to the widely varying patterns we find in our results, we believe that this is not sufficient for our findings to generalize beyond these specific datasets to the modalities that they represent. This points to a limitation of cross-modal fine-tuning work in general, which would benefit from a larger set of datasets, and in particular, more challenging tasks, as we find that the Satellite dataset is very simple.

Choice of pre-trained models. Our experiments focus on 1D tasks, for which we only experiment with encoder-only architectures (specifically RoBERTa-type models) even though other encoder-only models and even other architectures (e.g., encoder-decoder and decoder-only models) could also be used. We caution against claims about generalization of our results for these tasks to pre-trained models beyond just RoBERTa.

Ablating stage one. Our experiments focus on stages two and three of the ORCA pipeline, but stage one, i.e., the creation of the task-specific embedder and predictor, is not something we vary. In Shen et al. (2023) and in our work, the task-specific embedder consists of a convolutional layer, a layer norm, and a positional embedding, and the predictor consists of a linear projection. It would be interesting to test a much simpler method of converting dimensions in the embedder than a convolutional architecture, e.g., a linear projection, which we leave to future work.

Evaluating what is being transferred. In Section 5, we show that pre-training is necessary for some cross-modal transfer, but we still do not know exactly what is being transferred. The cross-modal transfer literature posits that pre-trained knowledge is somehow exploited in downstream tasks, but since we do not know how to quantify "knowledge" in this setting, we cannot make this claim. It is just as plausible that models pre-trained on tokens beyond a certain scale find better, more general solutions that are a good initialization for adapting to a new task. One way to further probe the transfer hypothesis would be by limiting the number of parameters that are allowed to change during fine-tuning, e.g., by using parameter-efficient fine-tuning with LoRA. We leave an exploration of this to future work.

B Dataset details

Table 1 shows the target and original proxy datasets considered, along with their dimension, type, number of classes, and the metric used to measure target task performance. The tasks are classified into two types, taking into account whether the task's output is a singular prediction (point) or multiple predictions (dense). The target datasets are described in more detail below.

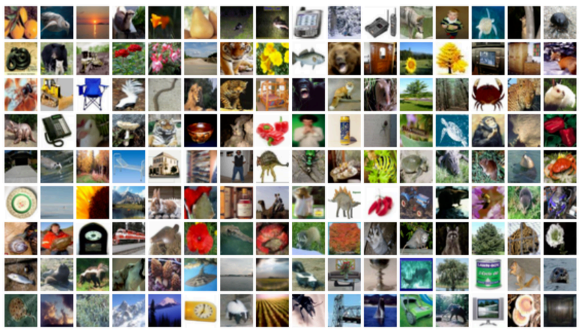


Figure 6: CIFAR-100 examples.

CIFAR-100: Standard Image Classification. (Alex, 2009) The dataset consists of 32x32 color images divided into 100 classes, based on the object represented by the image. Some examples can be seen in Figure 6.

Darcy Flow: Solving Partial Differential Equations (PDEs). (Li et al., 2020) The only regression task considered. Although, for the training stages, the dataset is divided into a total of 10 inferred classes. The dataset consists of 2D grids specifying the initial conditions of a fluid, as an output the same 2D grid on a later time is predicted.

DeepSEA: Predicting Functional Effects From Genetic Sequences. (Feingold et al., 2004) The dataset consists of a collection of genomic profiles to estimate the behavior of chromatin proteins,

Dim.	Target dataset	Type	Metric	# classes	Proxy dataset	# classes
	NinaPro	Point	0-1 error (↓)	18		
2D	CIFAR-100	Point	0-1 error (↓)	100	CIFAR-10	10
	Darcy Flow	Dense	relative l_2 (↓)	10		
	Satellite	Point	0-1 error (↓)	24		
1D	DeepSEA	Point (multi-label)	1 - AUROC (↓)	36	CoNLL-2003[3]	7
	ECG	Point	1 - F1 (↓)	4		

Table 1: Target datasets of each type along with the proxy datasets used for them in ORCA (Shen et al., 2023)

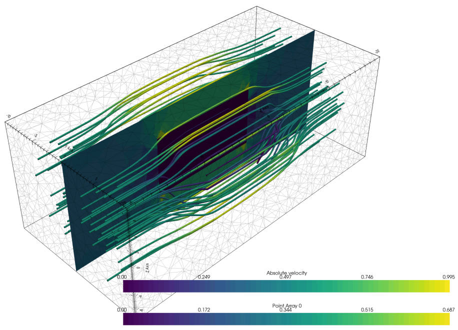


Figure 7: Example from the Darcy Flow dataset.

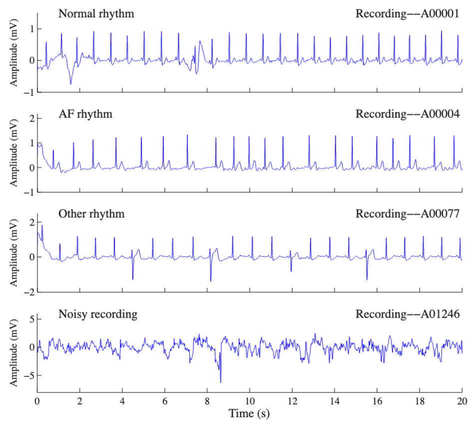


Figure 8: Examples of ECG recordings of the 4 different classes

classifying it into 36 classes.

ECG: Detecting Heart Disease. (Clifford et al., 2017) The dataset is formed by recordings of up to a minute of Electrocardiograms classified into four classes: normal, disease, other, or noisy rhythms. Figure 8 shows an example of each of the classes.

NinaPro: Classifying Electromyography Signals. (Atzori et al., 2012) A subset of NinaPro BD5 is taken, to classify the electromyography (sEMG) signals of a collection of hand movements in 18 classes. Some examples of the movements can be seen in Figure 9.

Satellite: Satellite Image Time Series Analysis. (Petitjean et al., 2012) The dataset consists of satellite image time series (SITS), tracking the land changes over the years, classifying them into 24 land cover types.

[3]We were unable to replicate the exact workflow to create the language features passed to the model, so we used the ones provided in the original ORCA GitHub.

C Embedder and predictor details

As described in Figure 1, in the first stage of the ORCA workflow (Shen et al., 2023), a task-specific embedder and predictor are created to support any combination of input-output dimensions. Throughout all our experiments, we kept the same architectures used in the original paper, which we will explain in this section.

Task-specific Embedding Network The architecture is composed of a convolutional layer with an input channel of the target dataset and an output channel of the dimension of the pre-trained model embedding space. The kernel size and stride can be treated as a hyperparameter, but in all our experiments for the 2D tasks both are set to four and, for the 1D tasks, are computed based on the input and target sequence length. After this, a layer norm and a positional embedder are added to obtain the final representation.

Task-specific Predictor Given the diversity of the tasks considered, two different architectures are implemented depending on the target task type. For

Algorithm 1 Efficient approximation of OTDD using class-wise subsampling from (Shen et al., 2023)

Input: target dataset $\{x^t, y^t\}$, number of target classes K^t, source dataset $S = \{x^s, y^s\}$, subsample size b, subsample round R

for each class $i \in [K^t]$ in the target dataset **do**
  Compute class weight $w_i = \frac{number\ of\ target\ data\ in\ class\ i}{total\ number\ of\ target\ data}$
  Generate data loader D_i consisting of data in class i
end for
for $i \in [K^t]$ **do**
  for $r \in [R]$ **do**
    Subsample b target data points D_{ir} uniformly at random from D_i
    Compute class-wise distance $d_{ir} = OTDD(D_{ir}, S)$
  end for
  Approximate class-wise OTDD by $d_i = \frac{1}{R}\sum_{i=1}^{R} d_{ir}$
end for
Approximate OTDD by $d = \sum_{i=1}^{K^t} w_i \dot{d}_i$

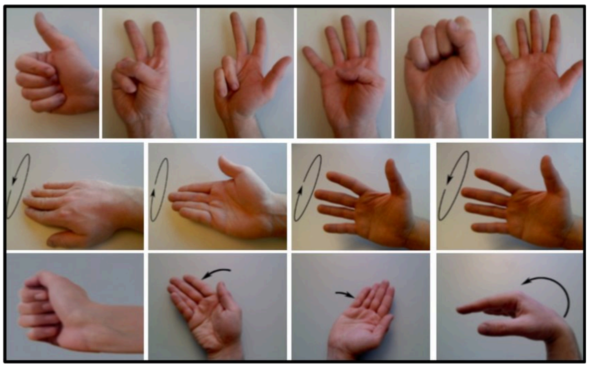


Figure 9: Samples of movements in NinaPro BD5 (Shen et al., 2019), the dataset contains the electromyography signals of the movements.

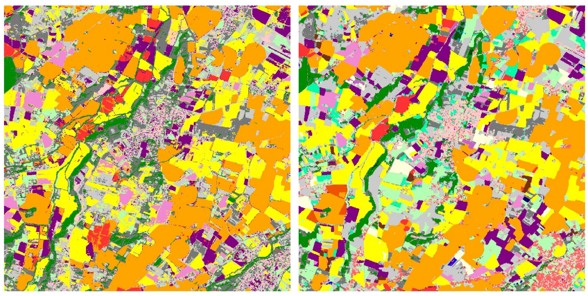


Figure 10: Example of Satellite (Petitjean et al., 2012)

the point tasks, average pooling along the sequence length dimension is applied, to obtain 1D tensors with the same length as the dimension of the pretrained model embedding space. Then to map to the number of classes of the target dataset, a linear layer is used. For dense tasks, a linear layer is applied to the sequence outputs to adjust the tensor shape. Then, this tensor is molded to the desired output dimension.

D OTDD approximation implementation

Following the original ORCA implementation (Shen et al., 2023), we also used an approximation of OTDD using class-wise subsampling, as described in Algorithm 1.

As described in the original paper, to tackle potential memory issues when computing OTDD, the dimensionality of the feature vectors is reduced by taking the average along the sequence length dimension. On top of that, the target dataset is divided into subsets based on the labels, each of these subsets will be approximated with the average of batch samples (the number of maximum samples taken from each class is determined for every dataset). Then the OTDD between each class representative and a sample of the proxy dataset (5000 samples for CIFAR-10 and 2000 for CONLL 2003) is computed. Finally, the overall OTDD is approximated by the weighted sum of the OTDD of all the classes in the task dataset.

E Experimental Details

We run our experiments using a single 80GB NVIDIA A100 GPU. As in the original paper (Shen et al., 2023), we implemented the base models using the Huggingface Transformers library (Wolf et al., 2020).

F Details on pre-trained RoBERTa models

Table 2 provides information about the amount of training data seen by the different RoBERTa variants released by Warstadt et al. (2020).

Model	Training data
roberta-base	~30B
roberta-base-1B-2	1B
roberta-base-100M-3	100M
roberta-base-10M-3	10M
roberta-base-random	0

Table 2: Models for pre-trained knowledge comparison, and their training data in number of tokens.

Does Fine-tuning a Classifier Help in Low-budget Scenarios? Not Much

Cesar Gonzalez-Gutierrez, Audi Primadhanty, Francesco Cazzaro, Ariadna Quattoni
Universitat Politècnica de Catalunya, Barcelona, Spain
{cesar.gonzalez.gutierrez, audi.primadhanty, francesco.cazzaro}@upc.edu,
aquattoni@cs.upc.edu

Abstract

In recent years, the two-step approach for text classification based on pre-training plus fine-tuning has led to significant improvements in classification performance. In this paper, we study the low-budget scenario, and we ask whether it is justified to allocate the additional resources needed for fine-tuning complex models. To do so, we isolate the gains obtained from pre-training from those obtained from fine-tuning. We find out that, when the gains from pre-training are factored out, the performance attained by using complex transformer models leads to marginal improvements over simpler models. Therefore, in this scenario, utilizing simpler classifiers on top of pre-trained representations proves to be a viable alternative.

1 Introduction

In the past few years, a dominating paradigm has emerged in text classification, primarily centered on a two-step approach: inducing pre-trained weights, followed by task fine-tuning using a transformer model with supervised labeled data (Radford et al., 2018; Devlin et al., 2019). The new approach has led to significant improvements over previous classification strategies based on simpler linear models trained on sparse bag-of-words feature representations.

The improvements observed in performance are often attributed to the induced representation (Miaschi and Dell'Orletta, 2020; Talmor et al., 2020; Xia et al., 2020). It is not surprising that leveraging contextual continuous word embeddings can lead to improvements by mitigating the sparsity issues of classical bag-of-words representations. At the same time, we expect that richer transformer architectures would enhance classification performance during fine-tuning. However, if the representation is already strong enough, is it justified to allocate additional resources for fine-tuning to achieve satisfactory results?

When the same architecture is shared for both pre-training and fine-tuning (Peters et al., 2018; Devlin et al., 2019), it becomes challenging to disentangle the relative influence of the representation and the classifier. To isolate the performance of each component, we propose an empirical study where we train both simple linear models and complex transformer models, with and without pre-trained representations, and test their performance in high and low annotation budget scenarios.

We specifically focus on investigating the previous question within the context of a low annotation budget scenario, where the availability of labeled data for fine-tuning is limited.

Our empirical study shows that:

- In low-budget scenarios, the incorporation of pre-trained representations results in a more significant performance improvement compared to high-budget scenarios. Moreover, when we isolate the gains attributed to pre-training, the performance gains of transformers over simpler models become marginal, meaning that the quality of the representations is the most important component.

- In this setting, a simple classifier on top of a contextual representation achieves competitive results compared to fine-tuning. Consequently, the impact of the classifier proves to be rather minimal, allowing us to utilize more cost-effective alternatives.

2 Related Work

While transformer (Vaswani et al., 2017) architectures are known to benefit from large amounts of training data for optimal performance (Ezen-Can, 2020; Kirstain et al., 2022), the pre-training plus fine-tuning approach has also shown promising results in low annotation budget scenarios (Ein-Dor

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 17–24
June 20, 2024

et al., 2020; Tamkin et al., 2022; Shelmanov et al., 2021; Zhang et al., 2022).

Fine-tuning is thought to adjust the pre-trained representations in order to simplify the downstream task (Zhou and Srikumar, 2022, 2021). However, the fine-tuning step itself can be unstable (Mosbach et al., 2021; Zhang et al., 2021) and sensitive to weight initialization (Dodge et al., 2020). These issues are particularly pronounced in low-budget scenarios (Margatina et al., 2022). To address these challenges, researchers have explored techniques such as parameter reduction (Han et al., 2021; He et al., 2021; Liu et al., 2018) or modifications to the fine-tuning procedure (Hua et al., 2021; Yang and Ma, 2022). Other authors have explored the possibility of using pre-trained representations directly with simpler classifiers (Li et al., 2021; Dubey et al., 2018)

The importance of representation choice has lately received a significant amount of attention from the active learning (AL) community (Schröder and Niekler, 2020; Zhang et al., 2017; Ein-Dor et al., 2020; Yuan et al., 2020; Yauney and Mimno, 2021; Margatina et al., 2022; Shelmanov et al., 2021). Most of the research in AL attempts to quantify what representation is best when training the initial model for active learning, which is usually referred to as the cold start problem (Lu and MacNamee, 2020; Zhang et al., 2022). The importance of word embeddings has also been studied in the context of highly imbalanced data scenarios (Sahan et al., 2021; Naseem et al., 2021; Hashimoto et al., 2016; Kholghi et al., 2016).

The main difference between our work and previous literature is that in prior studies, the fine-tuning process involved the simultaneous updates of both the pre-trained weights and the classifier, without considering their relative importance. Having established the relevance of the representation, especially in few-shot learning scenarios, we aim to investigate whether fine-tuning complex architectures in classification tasks is justified.

3 The Role of the Classifier in Low-budget Scenarios

To conduct our study, we aim to compare the performance of a transformer-based model and a simple classifier, trained with and without pre-trained representations. The main focus of our investigation will be on scenarios with a limited annotation budget, by utilizing learning curves. Each point in these curves represents a specific training size, enabling us to evaluate the model's performance as the data size increases. Additionally, we will report performance on the full dataset for the different models. Next, we detail the models, datasets, and learning curves employed.

Dataset	Size	Prior	Len.	Task
IMDB	50K	50%	313	sentiment
WiTox	224K	9%	78	toxicity
S140	1.6M	50%	23	sentiment
CivCom	2M	8%	58	toxicity

Table 1: Datasets statistics with the number of samples, target (positive) class prior, average token sequence length, and classification task.

3.1 Models

We contrast two model architectures: a transformer (BERT) and a max-entropy model (MaxEnt). Each of the models will be trained in two settings: 1) without pre-trained representations and 2) with pre-trained representations.

BERT (Devlin et al., 2019): $\text{BERT}_{\text{BASE-uncased}}$ model (110M parameters) using standard pre-training (BooksCorpus plus Wikipedia) and implemented using the HuggingFace Transformers library (Wolf et al., 2020). Learning without pre-trained representations means learning with randomly initialized weights (similar to Voita and Titov, 2020 and Zhang and Bowman, 2018). The hyper-parameter values can be found in A.2.

MaxEnt: A standard max-entropy model trained with l_2 regularization. When training without pre-trained representations, we used a sparse bag-of-n-grams representation. For the models with pre-training, we extracted static representations from the second-to-last hidden layer (Bommasani et al., 2020; Devlin et al., 2019) using the average of BERT's token embeddings (768 dimensions vectors). Our preliminary experiments have shown that such embeddings yield better performance than using BERT's `[CLS]` token (similar to the ablation studies in Devlin et al., 2019 and the observations presented in Lu and MacNamee, 2020). The regularization parameters and the optimal n-gram size were validated via 5-fold cross-validation.

3.2 Datasets

We use four textual classification datasets with both balanced and imbalanced label distributions,

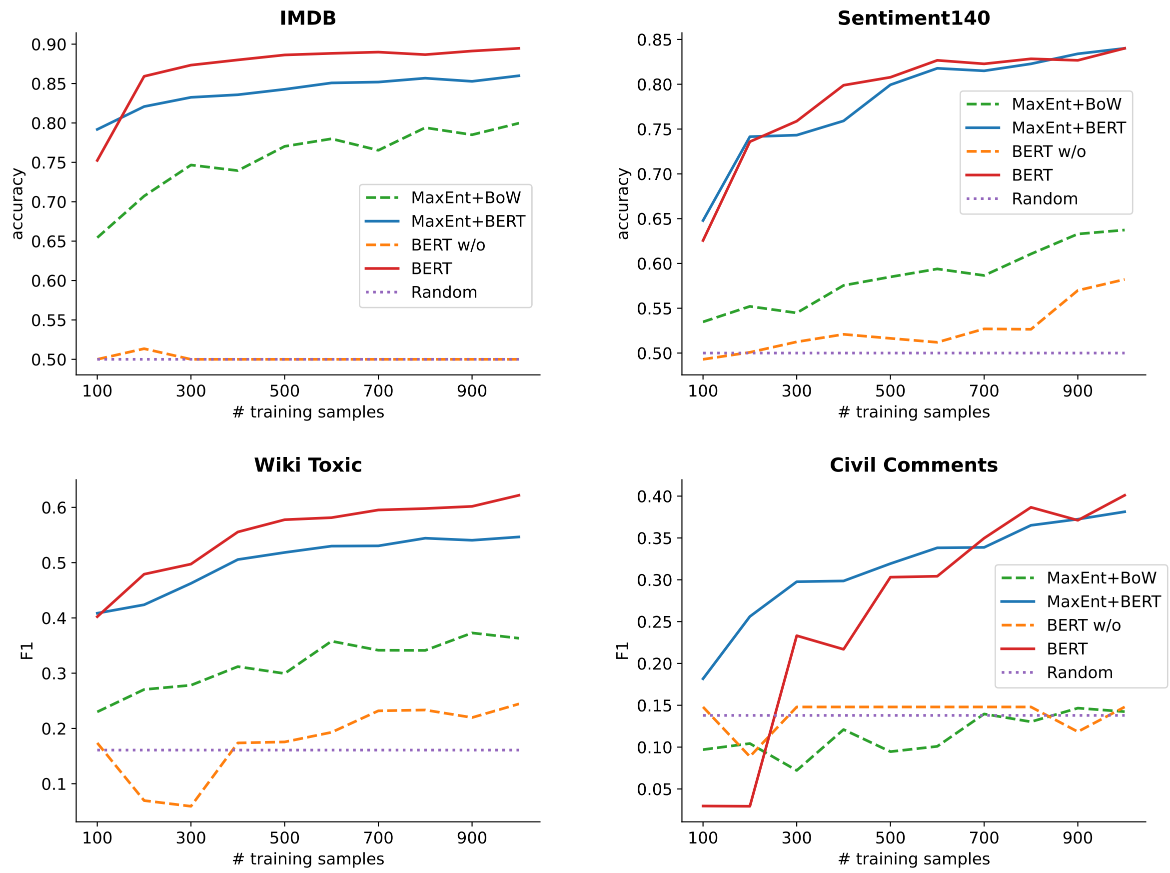


Figure 1: Performance of different models when learning with a limited annotation budget on various datasets. 'w/o' means without pre-trained representations. We also report the expected performance of a random classifier predicting i.i.d. labels.

encompassing two significant classification tasks (sentiment analysis and toxicity detection) across a variety of language registers and input lengths:

IMDB (Maas et al., 2011): Movie reviews annotated with sentiment labels. This is a dataset with a balanced distribution of labels.

Wikipedia Toxicity (WiTox; Wulczyn et al., 2017): Wikipedia discussion comments annotated with toxicity labels. This is a dataset with a highly imbalanced label distribution: less than 10% of the labels correspond to toxic comments.

Sentiment140 (S140; Go et al., 2009): A balanced dataset of Twitter messages annotated with sentiment.

Civil Comments (CivCom; Borkan et al., 2019): Opinions posted in the Civil Comments platform annotated for toxic behavior. This dataset exhibits a significantly skewed label distribution, with less than 10% toxic comments.

For Wikipedia Toxicity and Civil Comments, we have applied a pre-processing consisting of removing all markup code and non-alpha-numeric characters except relevant punctuation. Table 1 presents the datasets' summary statistics.

3.3 Learning curves

For our study, we generate learning curves where each point corresponds to a different training size with a budget of N samples. We create training sets by selecting the N random samples incrementally. N ranges from 100 to 1000 in increments of 100. At each step, new samples are added to the existing selection.

For every model, some hyper-parameters need optimization. At every point N in the learning curve, we create an 80/20% 5-fold cross-validation split and validate the optimal hyper-parameters. We then use these hyper-parameters to train a model using all the N training samples, and its performance is evaluated on the test set.

We repeat the experiments using 5 training sets and initializing the parameters using different random seeds. We report the mean results. As evaluation metrics we use: accuracy for the balanced

Dataset	Model	ALC w/o	ALC p.r.
IMDB	MaxEnt	0.75	0.84
	BERT	0.50	**0.87**
WiTox	MaxEnt	0.32	0.50
	BERT	0.18	**0.51**
CivCom	MaxEnt	0.11	**0.32**
	BERT	0.15	0.26
S140	MaxEnt	0.58	**0.79**
	BERT	0.53	**0.79**

Table 2: Model performance with a limited annotation budget, using pre-trained representations (p.r.) and without (w/o). We report the area under the learning curve (ALC) from 100 to 1000 examples, using accuracy for balanced datasets and F1 (of the target class) for imbalanced datasets. The best model performance for each dataset is reported in bold.

Dataset	Model	Performance w/o	Performance p.r.
IMDB	MaxEnt	0.89	0.89
	BERT	0.53	**0.93**
	Random	0.50	0.50
WiTox	MaxEnt	0.66	0.61
	BERT	0.48	**0.68**
	Random	0.16	0.16
CivCom	MaxEnt	0.60	0.57
	BERT	0.15	**0.70**
	Random	0.14	0.14
S140	MaxEnt	0.81	**0.86**
	BERT	0.77	**0.86**
	Random	0.50	0.50

Table 3: Model performance using all training data.

datasets (IMDB and Sentiment140) and F1 (of the target class) for the imbalanced datasets (Wikipedia Toxicity and Civil Comments).

In total, we performed 400 experiments for each model: 4 datasets, with and without pre-trained representations, 5 seeds, and 10 learning points. For BERT, the computation of each learning point took 27 minutes on average on a single Nvidia V100 GPU, totaling 177 hours of GPU computation. A.1 contains further details about the running times.

4 Results

Figure 1 shows our main results on analyzing the performance of models under the low-budget annotation setting. To summarize the learning curve results, we also compute a single performance score for each model: the area under the learning curve (ALC). This provides us with a more robust metric to compare the different models for a dataset[1]. Table 2 shows the results obtained.

We observe that in the low-budget scenario when pre-trained representations are used, the choice of model seems to be of little importance. Both the complex transformer model and a simple linear max-entropy model perform similarly.

In addition, when only very few labels are available (first curve points in Figure 1), the simpler model seems to outperform the more complex one. MaxEnt demonstrates a more stable behavior within this range, due to its fewer number of parameters. This shows that when the training set is small there is not much to be gained from fine-tuning all the layers of the model.

The biggest difference in performance in the low-budget scenario comes from the representation and not the architecture. In fact, without pre-trained representations, the more complex models perform significantly worse than simpler models. Pre-trained representations seem to be capturing some properties of the input space that can be exploited by all models. We suppose that since pre-training implicitly induces a distance space over words, models using pre-trained representations generalize more easily to unseen words. This would explain why pre-trained representations are especially helpful in the low-annotation budget scenario since generalization to unseen words is critical in this case.

Table 3 presents the performance results obtained by employing the entire training set. Within this data-rich scenario, typically used for model comparison, we first confirm the well-established fact that BERT with pre-trained weights yields better results than simpler models (Devlin et al., 2019). Interestingly, in this context, simpler models do not seem to obtain significant benefits from the use of pre-trained representations. Unlike the low-budget scenario, in this setting, fine-tuning all layers of the model results in significant performance improvements.

[1]Direct performance comparison across datasets is not always feasible because the underlying score may vary.

5 Conclusion

In this paper, we studied classifiers in a low-budget scenario, analyzing the impact of fine-tuning on performance by separating the benefits derived from pre-training weights from those of architectural fine-tuning.

Based on our findings, we recommend testing simple models that incorporate pre-trained representations before investing resources in fine-tuning complex models. In fact, when labeled data is scarce, the role of the representations is crucial, and the use of pre-trained representations enhances performance across all models, regardless of their complexity. As a result, the choice of classifier becomes irrelevant in this context compared to the quality of the representations. The marginal performance gains offered by more sophisticated architectures may not justify the additional computational resource demands.

Limitations

When studying the performance of a simple classifier over pre-trained representations, we have considered BERT as the representative for transformer-based models. A comparison with other transformer models, with a different number of parameters and embedding representations, would make our conclusions more general.

Our analysis is limited to binary classification tasks. Future research should aim to extend our study to other types of tasks to better understand the broader implications of our findings.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 853459. The authors gratefully acknowledge the computer resources at ARTEMISA, funded by the European Union ERDF and Comunitat Valenciana as well as the technical support provided by the Instituto de Física Corpuscular, IFIC (CSIC-UV). This research is supported by a recognition 2021SGR-Cat (01266 LQMC) from AGAUR (Generalitat de Catalunya).

References

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced Metrics for Measuring Unintended Bias with Real Data for Text Classification. *arXiv:1903.04561 [cs, stat]*. ArXiv: 1903.04561.

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.

Abhimanyu Dubey, Otkrist Gupta, Ramesh Raskar, and Nikhil Naik. 2018. Maximum-entropy fine grained classification. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

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 the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online. Association for Computational Linguistics.

Aysu Ezen-Can. 2020. A comparison of LSTM and BERT for small corpus. *CoRR*, abs/2009.05451.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter Sentiment Classification using Distant Supervision. *CS224N project report, Stanford*, 1(12):6.

Wenjuan Han, Bo Pang, and Ying Nian Wu. 2021. Robust transfer learning with pretrained language models through adapters. 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 854–861, Online. Association for Computational Linguistics.

Kazuma Hashimoto, Georgios Kontonatsios, Makoto Miwa, and Sophia Ananiadou. 2016. Topic detection using paragraph vectors to support active learning in systematic reviews. *Journal of Biomedical Informatics*, 62:59–65.

Ruidan He, Linlin Liu, Hai Ye, Qingyu Tan, Bosheng Ding, Liying Cheng, Jiawei Low, Lidong Bing, and Luo Si. 2021. On the effectiveness of adapter-based tuning for pretrained language model adaptation. 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 2208–2222, Online. Association for Computational Linguistics.

Hang Hua, Xingjian Li, Dejing Dou, Chengzhong Xu, and Jiebo Luo. 2021. Noise stability regularization for improving BERT fine-tuning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3229–3241, Online. Association for Computational Linguistics.

Mahnoosh Kholghi, Lance De Vine, Laurianne Sitbon, Guido Zuccon, and Anthony Nguyen. 2016. The Benefits of Word Embeddings Features for Active Learning in Clinical Information Extraction. In *Proceedings of the Australasian Language Technology Association Workshop 2016*, pages 25–34, Melbourne, Australia.

Yuval Kirstain, Patrick Lewis, Sebastian Riedel, and Omer Levy. 2022. A few more examples may be worth billions of parameters. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1017–1029, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Linyang Li, Demin Song, Ruotian Ma, Xipeng Qiu, and Xuanjing Huang. 2021. Knn-bert: Fine-tuning pre-trained models with knn classifier.

Liyuan Liu, Xiang Ren, Jingbo Shang, Xiaotao Gu, Jian Peng, and Jiawei Han. 2018. Efficient contextualized representation: Language model pruning for sequence labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1215–1225, Brussels, Belgium. Association for Computational Linguistics.

Jinghui Lu and Brian MacNamee. 2020. Investigating the effectiveness of representations based on pretrained transformer-based language models in active learning for labelling text datasets. *arXiv preprint arXiv:2004.13138*.

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.

Katerina Margatina, Loic Barrault, and Nikolaos Aletras. 2022. On the importance of effectively adapting pretrained language models for active learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 825–836, Dublin, Ireland. Association for Computational Linguistics.

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.

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*.

Usman Naseem, Matloob Khushi, Shah Khalid Khan, Kamran Shaukat, and Mohammad Ali Moni. 2021. A Comparative Analysis of Active Learning for Biomedical Text Mining. *Applied System Innovation*, 4(1):23.

Matthew E. 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.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training.

Marko Sahan, Vaclav Smidl, and Radek Marik. 2021. Active Learning for Text Classification and Fake News Detection. In *2021 International Symposium on Computer Science and Intelligent Controls (ISCSIC)*, pages 87–94. IEEE Computer Society.

Christopher Schröder and Andreas Niekler. 2020. A Survey of Active Learning for Text Classification using Deep Neural Networks. ArXiv:2008.07267 [cs] version: 1.

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.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Alex Tamkin, Dat Nguyen, Salil Deshpande, Jesse Mu, and Noah Goodman. 2022. Active learning helps pretrained models learn the intended task.

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

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.

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.

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*, WWW '17, pages 1391–1399, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Patrick Xia, Shijie Wu, and Benjamin Van Durme. 2020. Which *BERT? A survey organizing contextualized encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7516–7533, Online. Association for Computational Linguistics.

Chenghao Yang and Xuezhe Ma. 2022. Improving stability of fine-tuning pretrained language models via component-wise gradient norm clipping. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4854–4859, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Gregory Yauney and David Mimno. 2021. Comparing text representations: A theory-driven approach. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5527–5539, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, Online. Association for Computational Linguistics.

Kelly Zhang and Samuel Bowman. 2018. Language modeling teaches you more than translation does: Lessons learned through auxiliary syntactic task analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 359–361, Brussels, Belgium. 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*.

Ye Zhang, Matthew Lease, and Byron Wallace. 2017. Active Discriminative Text Representation Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022. A survey of active learning for natural language processing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6166–6190, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yichu Zhou and Vivek Srikumar. 2021. DirectProbe: Studying representations without classifiers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5070–5083, Online. Association for Computational Linguistics.

Yichu Zhou and Vivek Srikumar. 2022. A closer look at how fine-tuning changes BERT. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1046–1061, Dublin, Ireland. Association for Computational Linguistics.

A Appendix

A.1 BERT Runtime

Table 4 shows BERT's training plus testing running times for the budgets considered in the learning curves studied in this work. These experiments were performed using a single Nvidia V100 GPU.

Budget	IMDB	WiTox	CivCom	S140
100	15:07	38:02	27:30	01:00
200	17:24	40:17	27:31	01:21
300	19:57	41:54	27:45	01:45
400	21:35	42:27	29:57	02:18
500	22:54	43:02	31:35	03:03
600	25:19	47:09	32:28	03:24
700	26:50	44:50	34:02	03:35
800	28:48	48:09	34:59	03:40
900	31:02	48:40	35:45	04:01
1000	30:35	56:12	36:20	04:15

Table 4: BERT training and testing average runtime.

Table 5 displays the average speed of embedding generation, measured in samples per second.

Dataset	Gen. Time
IMDB	37.48 i/s
Sentiment140	135.42 i/s
Wiki Toxic	186.20 i/s
Civil Comments	131.37 i/s

Table 5: Embedding generation average speed.

Compared to fine-tuning, embedding extraction is a significantly more efficient operation and can feasibly be computed on the CPU.

A.2 Experimental Details

Table 6 contains a summary of BERT hyper-parameters used in the experiments.

Hyper-parameter	Value
Max. training epochs	10
Learning rate	$5 \cdot 10^{-5}$
AdamW λ	0.0
AdamW β_1	0.9
AdamW β_2	0.999
Attention dropout	0.1
Hidden dropout	0.1
Mixed Precision	fp16
Seq. length (IMDB)	350
Seq. length (Wiki Toxic)	150
Seq. length (Civil Comments)	150
Seq. length (Sentiment140)	50
Batch size (IMDB)	20
Batch size (Wiki Toxic)	50
Batch size (Civil Comments)	50
Batch size (Sentiment140)	64

Table 6: BERT hyper-parameters.

How Well Can a Genetic Algorithm Fine-tune Transformer Encoders? A First Approach

Vicente Ivan Sanchez Carmona and **Shanshan Jiang** and **Bin Dong**
Ricoh Software Research Center (Beijing) Co., Ltd
{Vicente.Carmona, Shanshan.Jiang, Bin.Dong}@cn.ricoh.com

Abstract

Genetic Algorithms (GAs) have been studied across different fields such as engineering or medicine to optimize diverse problems such as network routing, or medical image segmentation. Moreover, they have been used to automatically find optimal architectures for deep neural networks. However, to our knowledge, they have not been applied as a weight optimizer for the Transformer model. While gradient descent has been the main paradigm for this task, we believe that GAs have advantages to *bring to the table*. In this paper, we will show that even though GAs are capable of fine-tuning Transformer encoders, their generalization ability is considerably poorer than that from Adam; however, on a closer look, GAs ability to exploit knowledge from 2 different pretraining datasets surpasses Adam's ability to do so.

1 Introduction

Genetic Algorithms (GAs), a set of optimization methods, although widely studied in other fields such as electric engineering (Li and Ge, 2009; Sainath et al., 2021) or medicine (Ghosh et al., 2016), have not played a big role in the field of NLP as gradient descent algorithms. Indeed, a disadvantage of GAs is high running times when the search space is big. However, GAs possess advantages that make us reconsider their usefulness in NLP such as 1) algorithmic simplicity, 2) no vanishing- or exploding-gradient problems since no gradient signal is necessary, and 3) any mathematical expression can be optimized, such as Accuracy.

On the other hand, gradient descent approaches such as Adam (Kingma and Ba, 2015) are widely used not only due to the high fine-tuning scores they achieve for NLP models, but also due to a common –and barely challenged– assumption that prevails in the NLP field: fine-tuning a Transformer encoder that has been pretrained on two datasets will lead to considerably better scores than fine-tuning an encoder that was pretrained on either of the two pretraining datasets since, according to scaling laws (Kaplan et al., 2020), it is assumed that the former encoder has learned linguistic and (or) world knowledge from the two datasets, as opposed to the latter encoder which has only acquired knowledge from one dataset. However, we pose some skepticism on Adam's ability to efficiently exploit hidden knowledge from the 2 pretraining datasets encoded in such encoders.

In this paper, we propose a two-sided study of the ability of a GA to fine-tune pretrained Transformer encoders. Firstly, we study how well a GA can fine-tune pretrained encoders for the task of sentiment analysis across three datasets. And secondly, we put Adam to the test by comparing its ability to leverage pretrained knowledge from 2 pretraining datasets, at fine-tuning time, with respect to the ability of a GA to do so. Our main hypothesis is that the GA's crossover operator is the key factor to both fine-tune pretrained encoders and efficiently exploiting the knowledge from two pretraining datasets. To our knowledge, this is the first study of fine-tuning Transformer encoders via GAs.

Interestingly, our results are divided: we encountered both a negative and a positive result. On the one hand, although we confirm our hypothesis and show the ability of a GA to fine-tune Transformer encoders, we find two big deficiencies when compared against Adam: considerably higher training times (up to 46x) and a high drop in accuracy scores (up to 28 points) –a negative result. On the other hand, our results show that the GA outperforms Adam's ability to leverage knowledge from two pretraining datasets at fine-tuning time: fine-tuning encoders, pretrained on 2 datasets, via Adam leads to an average gain of 0.55 accuracy points with respect to fine-tuning encoders pretrained on only one pretraining dataset; but the GA's mean gain in performance under the same scenario is 1.65 points,

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 25–33
June 20, 2024

a relative increase of 200% (and up to 1540% for a particular case) –a positive result.

Overall, we believe GAs hold as an efficient mechanism for knowledge recombination of Transformer encoders. We hope the community will follow our work to carry out a deeper exploration of GAs on more challenging tasks.

2 Related Work

2.1 Genetic Algorithms

We note that our work is not the first to use GAs to optimize the weights of a Neural Network (NN) (Lander and Shang, 2015; Vázquez-Fernández et al., 2012; David and Greental, 2014). However, previous works evolved NNs (mainly feedforward NNs) containing only a few thousand weights. Our Transformer encoders contain almost 9.5 million parameters. On the other hand, recently, Sobhanam and Prakash (2023) used GAs for BERT-based models such as RoBERTa to automatically search for the best hyperparameter values for an optimal fine-tuning, such as the layers to be fine-tuned, the batch size, the learning rate, and the most suitable activation function; however, in that work, the GA is not used for finding the optimal weights of the model but only optimal hyperparameters.

2.2 AutoML

This area, also called Neural Architecture Search (Elsken et al., 2019), aims to automatically discover optimal architectures for deep NNs via variations of GAs. Recent works have shown the ability of GAs to find architectures as optimal as those from human designers (Miikkulainen et al., 2019; Xie and Yuille, 2017; Liang et al., 2019) and architectures that obtained SOTA results (Real et al., 2019). But, to our knowledge, there is no previous work where a Transformer model was fine-tuned using GAs.

3 Methods and Datasets

3.1 Genetic Algorithm

We use a variant of the Eclectic Genetic Algorithm (EGA) (Kuri and Quezada, 1998; Kuri-Morales et al., 2013). We chose it due to 1) its optimal trade-off between complexity,[1] efficiency, and memory usage due to GPU restrictions, and 2) its resemblance to an ideal GA (Mitchell et al., 1993). EGA follows the usual cycle of GAs. A population of n individuals (pretrained Transformer encoders in our case) is evolved through generations (an operation that can be cast as fine-tuning). In each generation, individuals are ranked by their fitness score (accuracy score on the train set) and crossed[2] to produce offspring (new encoders), and some of these offspring will experience mutation in their chromosomes (sets of hidden vectors). To allow EGA to cross encoders (recombine the knowledge encoded in their parameters), we replaced its crossover operator with the simulated-binary crossover: (Wirsansky, 2020):

$$child_1 = 0.5[(1+\beta)parent_1 + (1-\beta)parent_2]$$

$$child_2 = 0.5[(1-\beta)parent_1 + (1+\beta)parent_2]$$

where β is a hyperparameter manually chosen; and $parent_1, parent_2$ correspond to the set of all hidden vectors of two Transformer encoders. The crossover operation is done layer by layer[3] of both parent encoders which results in $child_1, child_2$ being the recombination of both parent encoders' vectors. Then these two offspring are evaluated on the train set, their fitness score is compared with that from all candidate encoders in the population, and the cycle repeats.

To test our hypothesis, we fix the crossover's probability of occurrence to $p_{cross} = 1$ to fully test its effect; we set the probability of mutation to $p_{mut} = 0.2$ to control for its effect. To mutate an encoder, we add a randomly drawn number in the [-1, 1] interval to randomly chosen weights.

3.2 Datasets

For pretraining encoders, we use 2 popular datasets: WikiText-103 (wiki) (Merity et al., 2017) and 1-Billion-Word (lm1b) (Chelba et al., 2013). For fine-tuning, we use three popular sentiment analysis datasets: SST-2 (Socher et al., 2013), IMDB (Maas et al., 2011), and Yelp (Zhang et al., 2015). We chose these downstream datasets for interpretability of results as binary accuracy scores are obtained.

[1]More complex than the Canonical GA (CGA) (Sivanandam and Deepa, 2008) but simpler than latest GAs. We note that we also experimented with the CGA, but we obtained poor results due to its over-simplicity which refrained it to cope with the high-dimensional space of Transformer models.

[2]The individual in rank i is crossed with the individual in rank $n - i + 1$, i.e. the best individual is crossed with the worst one and so on.

[3]For example, the first attention layers of two encoders will be crossed to produce two attention layers, one for each child.

4 Experiments and Results

4.1 Experimental Setup

We used the Transformer encoder variant from the KerasNLP framework (Watson et al., 2022). We pretrained 10 different encoders with each pretraining dataset by varying random seeds;[4] we refer to them as either wiki or lm1b encoders according to the dataset used. We also pretrained 5 different encoders using both pretraining datasets; we call them Mixed encoders. For some experiments with EGA we used randomly initialized encoders; we call them random encoders. We note 1) the same pretrained encoders are used for both cases fine-tuning them via Adam (baselines) and fine-tuning them via EGA, except for the Mixed encoders which are used only as baselines; 2) for all experiments with EGA, the number of generations is set to 100, the population size to 20 encoders; to obtain means and standard deviations, we run EGA 3 times for each sentiment analysis dataset using different random seeds; 3) for both Adam and EGA, to compute downstream mean scores we use validation or test accuracy scores (depending if the dataset has a test set) of the encoders with highest validation score.

4.1.1 Gains in Accuracy Score

For Adam, we define *gain in accuracy score* as the amount of performance increase in accuracy points obtained by fine-tuning encoders pretrained on 2 datasets with respect to the score obtained by fine-tuning encoders pretrained on only one of the two pretraining datasets. For example, the points increased by fine-tuning Mixed encoders with respect to fine-tuning wiki encoders on the SST-2 data. We refer to this gain as $gain_{Adam}$.

For EGA, we define gain in accuracy as the gain in points obtained by evolving (fine-tuning) wiki and lm1b encoders in the same population with respect to the score obtained by evolving only encoders of a single type (wiki or lm1b) which is the equivalent figure of comparing leveraging two pretraining datasets at fine-tuning time vs. only one dataset; we refer to this as $gain_{EGA}$.

We compute gains in accuracy score as follows:

$gain_{Adam} = acc_{Mixed\_enc} - acc_{single\_type\_enc}$

$gain_{EGA} = acc_{wiki+lm1b\_enc} - acc_{single\_type\_enc}$

where acc means accuracy and $single\_type\_enc$ refers to either wiki or lm1b encoders. To compare EGA's gains in performance with those from Adam, we compute the relative increase in gain provided by EGA:

$$\frac{gain_{EGA} - gain_{Adam}}{|gain_{Adam}|} \times 100\% \qquad (1)$$

4.1.2 Effect of Number and Type of Encoder

To fully test EGA's ability to recombine knowledge from encoders, we carry out experiments across 6 levels where we vary the number and type of pretrained encoders. At Level 1, populations consist of 10 different lm1b encoders and 10 random encoders; and similarly for Level 2 where instead of lm1b we use wiki encoders. Populations at Levels 3, 4, and 5 contain 5, 10, and 15 pretrained encoders, respectively; but, different from Levels 1 and 2, we use both lm1b and wiki encoders (50% wiki and 50% lm1b), and the rest of the population are random encoders. Finally, populations at Level 6 consist only of pretrained encoders (10 wiki and 10 lm1b).

4.2 Results

4.2.1 Baselines

Fine-tuning encoders pretrained on both datasets via Adam leads to two substantial gains in score on SST-2 data: 1.69 and 1.1 points (Tables 2 and 3) with respect to wiki and lm1b encoders, respectively, which are the differences in accuracy from Mixed encoders and wiki, lm1b encoders in Table 4. However, on the other downstream datasets, this leads to minor gains: a gain of 0.36 points for IMDB data when lm1b and Mixed encoders are measured against each other, and gains of 0.1 and 0.58 points for Yelp data (Tables 2 and 3). Surprisingly, we observe a drop in gain of 0.52 points for IMDB data (Table 2): wiki encoders achieve a superior accuracy score (85.03, Table 4) than Mixed encoders (84.51) (we provide a possible explanation for this finding in Section 5).

By averaging all gains in score from Adam in Tables 2 and 3, we observe that Mixed encoders lead to a mean increase of only 0.55 points. However, we do not jump straightaway to the conclusion that, at fine-tuning time, Adam's ability to leverage knowledge from encoders pretrained on 2 datasets is not as impactful as we expected since these results could be obscured by a *ceiling effect*, as we discuss in Section 5.

4.2.2 Genetic Algorithm Results

SST-2: As shown in Table 1, EGA's best score on SST-2 data (59.15 points) comes from a mixed

[4] We believe that by doing so the encoders can pick different patterns even if pretrained on the same data.

Dataset	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
SST-2	57.25(0.021)	57.71(0.010)	57.91(0.011)	59.15(0.003)	58.69(0.016)	57.98(0.020)
IMDB	60.09(0.001)	57.0(5e-4)	57.41(0.002)	57.06(0.001)	57.87(0.023)	59.59(0.015)
Yelp	56.82(0.008)	58.0(0.001)	59.43(0.044)	59.04(0.010)	59.64(0.014)	57.33(0.006)

Table 1: Mean validation or test accuracy scores across three different random seeds of fine-tuning pretrained encoders via EGA (standard deviations in parenthesis) for different Levels as described in Section 4.1.2.

Dataset	Adam	EGA	Relative increase
SST-2	1.69	1.44	-14.79%
IMDB	-0.52	2.59	598.07%
Yelp	0.1	1.64	1540%

Table 2: Gains in accuracy points by Adam and EGA: two pretraining datasets (wiki+lm1b) vs. wiki dataset. Column Relative increase shows the increase of performance of EGA over Adam as in Section 4.1.1.

Dataset	Adam	EGA	Relative increase
SST-2	1.1	1.9	72.72%
IMDB	0.36	-0.5	-238.88%
Yelp	0.58	2.82	386.20%

Table 3: Gains in accuracy points by Adam and EGA: two pretraining datasets (wiki+lm1b) vs. lm1b dataset. Column Relative increase shows the increase of performance of EGA over Adam as in Section 4.1.1.

Dataset	Wiki enc	Lm1b enc	Mixed enc
SST-2	75.12(0.014)	75.71(0.013)	76.81(0.010)
IMDB	85.03(0.006)	84.15(0.006)	84.51(0.003)
Yelp	87.62(0.006)	87.14(0.003)	87.72(0.004)

Table 4: Mean validation or test accuracy scores across encoders (standard deviations in parenthesis) of fine-tuning pretrained encoders via Adam.

population (Level 4): 5 wiki and 5 lm1b encoders (and the rest random encoders). On the other hand, the lowest accuracy scores come from populations where only one type of encoder is used: Level 1 (only lm1b) and Level 2 (only wiki). As we see, recombining the hidden knowledge from wiki and lm1b encoders leads to substantial gains of 1.44 and 1.9 points (Tables 2 and 3) compared to crossing either only wiki or only lm1b encoders. Comparing EGA's gains vs. Adam's gains, we see that Adam obtains a bigger gain than EGA when using two pretraining datasets as opposed to only wiki data, as shown in Table 2: 1.69 vs. 1.44 points; however, this figure turns around for lm1b encoders where EGA's increase in gain is superior to that of Adam by 72.72%.

IMDB: We observe a clear gain in performance when knowledge from wiki encoders is mixed with that from lm1b encoders: a rise of 2.59 points (Table 2) which is the difference between crossing only wiki encoders (Level 2, Table 1) and crossing both encoders type (Level 6). Compared to Adam's gain in score (-0.52 points) EGA achieves a superior relative increase in performance of 598%. Nevertheless, similar to Adam, we see a drop in gain: lm1b encoders provide better results for EGA than wiki+lm1b encoders by 0.5 points (Level 1 vs. Level 6) as shown in Table 3.

Yelp: The best scenario comes from mixing knowledge from both encoder types (Table 1, Level 5): 59.64 points, providing a gain in score of 1.64 points with respect to fine-tuning only wiki encoders (Level 2), representing a remarkable relative increase of performance of 1540% with respect to the gain obtained by Adam of only 0.1 points (Table 2). From Table 3 we see the biggest gain in accuracy score obtained by EGA across all sentiment analysis datasets: 2.82 points increase by, again, crossing wiki with lm1b encoders (Level 5) as opposed to only lm1b encoders (Level 1).

Fine-tuning times: As we see in Table 5, EGA takes considerably more time than Adam (at least 31 times more) representing a disadvantage.

Dataset	Adam	EGA	Factor
SST-2	1.57	51.59	33x
IMDB	3.42	158.92	46x
Yelp	2.9	90.13	31x

Table 5: Average time in min. that Adam and EGA take to obtain an encoder with the highest validation score, and the factor of difference between them.

5 Discussion and Conclusions

How well can EGA fine-tune Transformer encoders? We observed in Table 1 that EGA is able to fine-tune the encoders on the sentiment analysis datasets with scores reaching, or close to, the

60 points threshold. Although there is a wide gap compared to Adam's scores (up to 28 points), we believe these results show the capability of GAs for fine-tuning Transformer encoders.

Exploiting knowledge from pretraining datasets: Another important aspect of fine-tuning is leveraging pretrained knowledge from two datasets. We observed in Tables 2 and 3 how Adam more often than not achieves small gains in performance with an average gain across datasets of only 0.55 points. Remarkably, EGA better exploits the hidden knowledge from wiki and lm1b encoders by obtaining and average gain of 1.65 points (3 times Adam's gain). On a closer look, we observe substantial relative increases of performance from EGA of up to 1540% as shown for the Yelp dataset.

Caution must be applied: We interpret these results with precaution. We cannot firmly conclude that Adam's capability of leveraging knowledge from encoders pretrained on 2 datasets will invariably lead to such a small average gain in accuracy for any other task or dataset since we may be facing a ceiling effect (Cohen, 1995). This effect happens either when the NLP model is too close to a perfect score (which is not our case) or when the NLP model is too close to its maximum capability in solving a task, which may be our case. It is possible that the variant of Transformer encoder we used, when pretrained on only one dataset, is already close to the maximum score it can achieve; thus, knowledge from another pretraining dataset helps but not as much as it would for a more difficult task where the initial score is small enough to leave room for improvement.

2 is not always better than 1: We saw the interesting finding that fine-tuning encoders pretrained on only 1 dataset led to the best results for the IMDB dataset, for both Adam (via wiki encoders) and EGA (via lm1b encoders). To provide a plausible rationale, we manually reviewed IMDB, wiki, and lm1b instances to find any qualitative patterns. Our first observation is the similarity in writing style between IMDB and wiki instances: long texts with a clear description of an item, entity, or event supported by facts or arguments and followed by a conclusion –patterns that Adam may have recovered from wiki encoders. On the other hand, we notice the newspaper writing style in lm1b instances which somehow differs from those in IMDB; probably, EGA exploited factuality and cultural patterns from the news articles in lm1b that helped to classify IMDB instances since both datasets are contemporary with a short time gap. Moreover, we believe that the different patterns in wiki and lm1b instances, rather than complementing to each other to improve on the downstream scores, as in the case for the SST-2 and yelp data, they are at odds with each other for the IMDB dataset; however, it is unclear exactly in which way. We believe this finding requires a deeper analysis given the complexity of the IMDB instances (Otterbacher, 2013).

Future work: We delimited our work to a specific choice of Transformer encoder, Genetic Algorithm, pretraining data, and downstream task. Naturally, further experimentation is necessary to generalize our results, such as studying more complex GAs and hybrid approaches that take advantage of the strengths of both Adam (high scores in low time) and GAs (ability to exploit different pretraining data) for fine-tuning more complex NLP models such as BERT (Devlin et al., 2019); test on other pretraining datasets, such as the BookCorpus (Zhu et al., 2015) or C4 (Raffel et al., 2020); and test the hypotheses proposed in this work on more complex downstream tasks and datasets to either confirm our results and elaborate upon them, or to pinpoint possible ceiling effects.

Acknowledgements

We thank the anonymous reviewers for their insightful comments which helped improve the Introduction section and the interpretation of results of this work.

References

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *Interspeech*.

Paul R. Cohen. 1995. *Empirical methods for artificial intelligence*. MIT Press, Cambridge, MA, USA.

Omid E David and Iddo Greental. 2014. Genetic algorithms for evolving deep neural networks. In *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pages 1451–1452.

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.

Yadolah Dodge. 2008. *Coefficient of Variation*, pages 95–96. Springer New York, New York, NY.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2019. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(1):1997–2017.

Payel Ghosh, Melanie Mitchell, James A. Tanyi, and Arthur Y. Hung. 2016. Incorporating priors for medical image segmentation using a genetic algorithm. *Neurocomputing*, 195:181–194. Learning for Medical Imaging.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *ArXiv*.

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*.

Angel Kuri and Carlos Quezada. 1998. A universal eclectic genetic algorithm for constrained optimization. *Proceedings 6th European Congress on Intelligent Techniques & Soft Computing, EUFIT'98*.

Angel Fernando Kuri-Morales, Edwin Aldana-Bobadilla, and Ignacio López-Peña. 2013. The best genetic algorithm ii. In *Advances in Soft Computing and Its Applications*, pages 16–29, Berlin, Heidelberg. Springer Berlin Heidelberg.

Sean Lander and Yi Shang. 2015. Evoae–a new evolutionary method for training autoencoders for deep learning networks. In *2015 IEEE 39th Annual Computer Software and Applications Conference*, volume 2, pages 790–795. IEEE.

Taoshen Li and Zhihui Ge. 2009. A multiple qos anycast routing algorithm based adaptive genetic algorithm. In *2009 Third International Conference on Genetic and Evolutionary Computing*, pages 89–92.

Jason Liang, Elliot Meyerson, Babak Hodjat, Dan Fink, Karl Mutch, and Risto Miikkulainen. 2019. Evolutionary neural automl for deep learning. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 401–409.

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.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. *International Conference on Learning Representations*.

Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, and Babak Hodjat. 2019. Chapter 15 - evolving deep neural networks. In Robert Kozma, Cesare Alippi, Yoonsuck Choe, and Francesco Carlo Morabito, editors, *Artificial Intelligence in the Age of Neural Networks and Brain Computing*, pages 293–312. Academic Press.

Melanie Mitchell, John Holland, and Stephanie Forrest. 1993. When will a genetic algorithm outperform hill climbing. *Advances in neural information processing systems*, 6.

Jahna Otterbacher. 2013. Gender, writing and ranking in review forums: a case study of the imdb. *Knowledge and Information Systems*, 35:645–664.

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. *J. Mach. Learn. Res.*, 21(1).

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. 2019. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789.

G. Sainath, S. Vignesh, S. Siddarth, and G. Suganya. 2021. Application of neuroevolution in autonomous cars. In *International Virtual Conference on Industry 4.0*, pages 301–311, Singapore. Springer Singapore.

S.N. Sivanandam and S.N. Deepa. 2008. *Introduction to Genetic Algorithms*. Springer Berlin, Heidelberg.

H. Sobhanam and J. Prakash. 2023. Analysis of fine tuning the hyper parameters in roberta model using genetic algorithm for text classification. *International Journal of Information Technology*, 15:3669–3677.

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.

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.

Eduardo Vázquez-Fernández, Carlos A. Coello Coello, and Feliú D. Sagols Troncoso. 2012. Assessing the positional values of chess pieces by tuning neural networks' weights with an evolutionary algorithm. In *World Automation Congress 2012*, pages 1–6.

Matthew Watson, Chen Qian, Jonathan Bischof, François Chollet, et al. 2022. Kerasnlp. https://github.com/keras-team/keras-nlp.

Eyal Wirsansky. 2020. *Hands-On Genetic Algorithms with Python*. Packt Publishing, Birmingham, UK.

Lingxi Xie and Alan Yuille. 2017. Genetic cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1379–1388.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

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 *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

A Appendix

A.1 Population-based Analysis

In this subsection, we show an additional analysis at the population level across the last 4 levels, where populations consist of both wiki and lm1b encoders, for the SST-2 dataset. Figures 1, 2, 3, 4 show the evolution of accuracy scores through generations. In each figure, training curve_1, validation curve_1 correspond to the evolution of one population. Thus, we show the evolution of 5 populations per level where each population is evolved using a different random seed. In each curve, each point represents the average accuracy score of one population for a given generation number.

As we see across all figures, our belief that Transformer encoders pretrained on the same dataset, but using a different random seed for pretraining, can capture different linguistic or world knowledge seems to be supported by these plots since at the beginning of all evolution processes the standard deviations for each population are very wide, which means that accuracy scores across each individual vary to a great extent which seems to imply that individuals encode different knowledge (some of them having learned patterns more useful for the SST-2 data than others) which is reflected in their different chromosomes.

Also, we observe in Figures 1 and 2 that for Levels 3 and 4, around generation $gen = 40$, most of

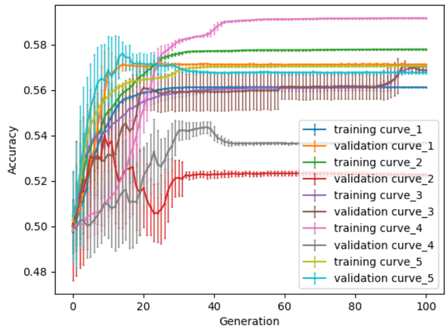


Figure 1: Average accuracy scores at the population level across generations for Level 3. Bars represent standard deviations.

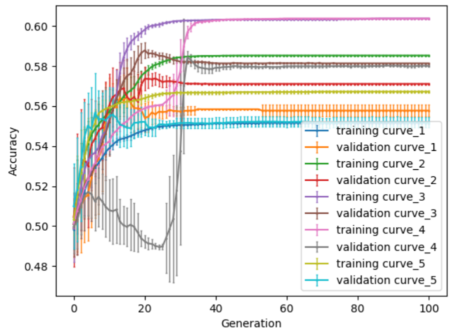


Figure 2: Average accuracy scores at the population level across generations for Level 4. Bars represent standard deviations.

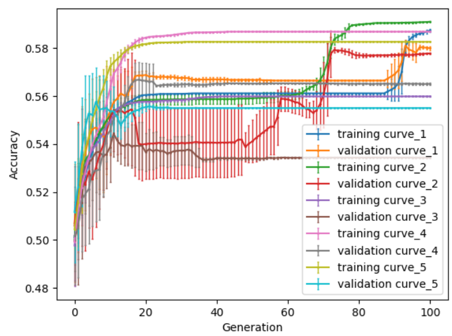


Figure 3: Average accuracy scores at the population level across generations for Level 5. Bars represent standard deviations.

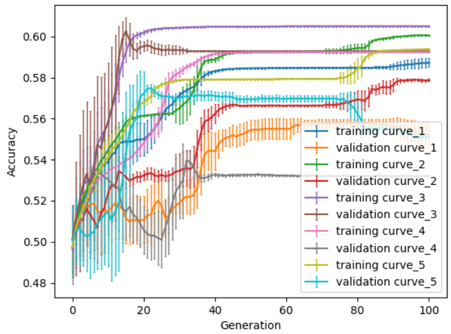


Figure 4: Average accuracy scores at the population level across generations for Level 6. Bars represent standard deviations.

the populations tend to converge to the final average accuracy, and in several cases the variation is minimal which means that individuals should share a large part of their genetic material with each other. This is a well-known effect in GAs and it tends to lead to local optima. However, for Levels 5 and 6 where most or all encoders are pretrained, stability for some populations tends to arrive at the last generations as there are cases where populations still see an increase in their average scores by almost the end of the run; this could mean that a bigger diversity of both wiki and lm1b encoders is helpful for avoiding or escaping local minima.

A.2 Genetic Analysis of the Best Individual

The best individual from all our experiments with the SST-2 dataset comes from Level 4; this encoder achieved a validation score of $val = 0.6135$. We traced back all its parents up to the first generation to have an idea of how its chromosome is formed. Not surprisingly, half of its genetic material is formed by weight vectors from wiki encoders and half from lm1b encoders. This piece of evidence further supports our hypothesis; recombining knowledge from different types of encoders leads to optimal individuals. It seems that weight vectors from different encoder types may encode different type of linguistic or world knowledge and when recombined they produce parameters more fit to the task at hand. We leave this hypothesis to be tested in future work.

A.3 Robustness to Variability

We measured how robust is each optimization method to the impact of random seed variation on the downstream scores; ideally, optimization methods would provide a robust estimate of the accuracy which translates into low variability. To measure this property, we computed the Coefficient of Variation (CV) (Dodge, 2008) since directly comparing the standard deviations from Adam and EGA is not a reliable approach due to the wide gap between mean accuracy scores from both methods. The coefficient of variation is a standardized measure that takes into account the size of the mean scores as follows:

$$CV = \frac{standard\_deviation}{mean} \times 100\% \qquad (2)$$

Thus, higher CV values represent a higher degree of variability. We compute coefficients of variation using means and standard deviations from Tables 4 and 1 for Adam and EGA, respectively. We find that while Adam's CV values range from 0.007% to 0.018%, EGA's CV values falls in the 0.0009%-0.07% range, both intervals containing extremely low signs of variation showing that both methods exhibit comparably high and robust estimates of accuracy.

A.4 Sampling of SST-2

To allow for a faster (and more environmentally friendly) training on the SST-2 data with EGA, we investigated if we could reduce its train set size through a learning curve. The learning curve in Figure 5 was obtained by evolving 20 randomly initialized encoders for 100 generations across 5 different random seeds. It shows that the best validation scores come from using approx. 5% of the train set (3072 instances).[5] Thus, we chose to use a random sample of size 3072 for our experiments with EGA and Adam.

A.5 Transformer Model and Training Details

Our target model is the Transformer encoder variant implemented in the KerasNLP framework which is roughly the equivalent of a half-size Transformer encoder from the original Transformer model in (Vaswani et al., 2017). We chose this variant mainly for memory consumption reasons when fine-tuning it with the genetic algorithm. We used same settings and hyperparameters as in (Watson et al., 2022) to have a fully reproducible baseline. Also, for some of our experiments we used the same dataset for pretraining (WikiText-103 dataset) and the same dataset for fine-tuning (SST-2) as

[5]We used up to approx. 88% of the dataset to keep it balanced between positive and negative labels since we are optimizing accuracy scores.

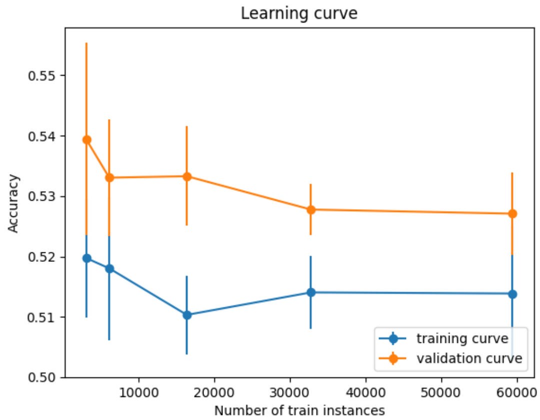


Figure 5: Learning curve. Each point is averaged on 5 different runs. Bars represent standard deviations.

those used in the KerasNLP original implementation.

More concretely, this Transformer implementation consists of 3 encoder blocks each with 4 attention heads; feedforward layer size is 512; token and learnable position embeddings are of dimension 256; sequence length of 128 tokens, and Word Piece Tokenizer. Total number of parameters is almost 9.5 million trainable weights. Pretraining batch size is 128, fine-tuning batch size is 32, sequence length is set to 128, mask rate is set to 0.25, dropout rate is set to 0.1, epsilon is set to $1e-5$, pretraining learning rate is $5e-4$, fine-tuning learning rate is $5e-5$, and pretraining epochs is set to 8. There is a parameter which we change from the original implementation; we increased the number of fine-tuning epochs to 15 since before the 15th epoch validation scores go down.

A.6 Hardware and Software Used

We used Tensorflow version 2.10.1, KerasNLP version 0.3.1, python version 3.10.9. To run our experiments we used an Nvidia RTX3060 GPU. The total time that all our experiments took to run was 495.55 hrs. (20.64 days).

I Have an Attention Bridge to Sell You: Generalization Capabilities of Modular Translation Architectures

Timothee Mickus **Raúl Vázquez** **Joseph Attieh**

University of Helsinki
firstname.lastname@helsinki.fi

Abstract

Modularity is a paradigm of machine translation with the potential of bringing forth models that are large at training time and small during inference. Within this field of study, modular approaches, and in particular attention bridges, have been argued to improve the generalization capabilities of models by fostering language-independent representations. In the present paper, we study whether modularity affects translation quality; as well as how well modular architectures generalize across different evaluation scenarios. For a given computational budget, we find non-modular architectures to be always comparable or preferable to all modular designs we study.

1 Introduction

Machine Translation (MT) has historically been under two influences that seem *a prima facie* contradictory. One of the goals of MT research is to provide means of converting sentences from any language to any other. On the one hand, generalization capabilities hinge on our systems producing language agnostic representations. On the other hand, MT models ought to be apt at encoding the specifics of source languages (Belinkov et al., 2017). The former of these trends has deeply marked this field—the concept of an 'interlingua' runs through most of the history of MT research, from Richens (1956) to Lu et al. (2018). The latter has recently motivated the development of modular approaches, where network parameters are specifically tied to a specific language.

How can we reconcile these two seemingly paradoxical trends? One promising approach is the inclusion of fully-shared subnetworks in modular architectures, and especially *bridge* components: They have been argued to foster language-independent representations (Zhu et al., 2020) as well as zero-shot generalization capabilities (Liao et al., 2021). Our aim is to carefully assess whether modular architectures in general and bridges do indeed foster greater generalization capabilities.

We therefore study six architectures, five of which modular, with a particular focus on how they generalize—both to unseen translation directions, and to novel domains. We find that modular systems still struggle to remain competitive with fully-shared MT systems in scenarios when not all translation directions are available—a conclusion that affects systems with and without fixed-size bridges equally. While encoder-sharing modular designs can rival or outperform non-modular settings in a wide range of scenarios, all other systems we study struggle in zero-shot and out-of-distribution conditions, strongly questioning that fully-shared sub-networks in modular MT systems can improve their generalization capabilities.

2 Related Work

The full span of multilingual NMT (MNMT) architectures rely in the implicit assumption that the systems leverage the multilingual data by creating a shared encoding space via sharing: from fully-shared models (Johnson et al., 2017), to fully-modular systems, where sharing occurs only at dataset level (Escolano et al., 2021). In this work, we assess those two extreme cases, focusing in the modular NMT systems that incorporate some parameter-sharing bridging layers. Lu et al. (2018) introduced an attentional neural interlingua, which processes language-specific encoder embeddings to produce language-agnostic representations. Zhu et al. (2020) proposed a language-aware interlingua that transforms the encoder representation to a shared semantic space, showcasing practical means of fostering the semantic consistency of translations. Vázquez et al. (2019) integrated a shared inner-attention mechanism, referred to as "attention bridge", based on the work of Lin et al. (2017), to generate fixed-size sentence representations. Fur-

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 34–40
June 20, 2024 ©2024 Association for Computational Linguistics

ther studies by Raganato et al. (2019) and Vázquez et al. (2020), whose work we specifically build upon, emphasized the advantages of using multiple attention heads on the semantic quality of the translation—as well as challenges, particularly with translating longer sentences. Boggia et al. (2023) explored the effects of sharing encoder parameters vs. increasing the number of languages in modular MNMT. More recently, Purason and Tättar (2022) used layers shared by language groups to enhance translation, Mao et al. (2023) proposed a variable-length bridge that uses a classification layer to predict its length, and in Pires et al. (2023) the encoder is built with interspersed fully-shared and language-specific layers.

3 Experimental Methodology

3.1 Model Variants

All the models we consider are Transformer-based (Vaswani et al., 2017), and implemented with the MAMMOTH library (Mickus et al., 2024).[1] An overview of the different modular architectures we consider is displayed in Figure 1. We ensure that all datapoints are processed by the same number of encoder and decoder layers (6 and 6 resp.).

Non-modular baseline. To provide a reasonable point of comparison with existing approaches, we consider a simple non-modular architecture where all parameters are shared across all translation directions. We note these fully-shared models as $\mathcal{F}$.

Fully modular baseline. A second natural point of comparison is a modular system without bridge; e.g. Escolano et al. (2021). Such models, noted $\mathcal{N}$ below, contain one 6-layer Transformer encoder and one 6-layer Transformer decoder per language, which are then selected for predictions depending on the desired language pair.

Semi-modular approaches. All other remaining architectures we will discuss contain both language specific and language-independent parameters. A simple means of achieving this consist in using a single shared encoder for all source languages (abbrv. $\mathcal{E}$), which would allow to leverage training signals from all source languages so as to provide more robust encoder representations. Conversely, one can consider employing a single shared decoder for al target languages (abbrv. $\mathcal{D}$) in the hopes of bolstering generation capabilities.

[1] Configuration files available at `github.com/Helsinki-NLP/mammoth/tree/main/examples/ab-neg/`.

Bridges. We also consider models with a "bridge" layer, i.e., where all parameters are language specific aside from the last Transformer layer in the encoder. Such models have been explored by e.g. Boggia et al. (2023). These models are noted $\mathcal{T}$, and contain 5-layer language-specific Transformer encoders, followed by a shared Transformer layer serving as a bridge—i.e. they are $\mathcal{N}$-type modular systems where the parameters of the last layers of each encoder are tied.

Fixed-size attention bridges. An alternative proposed by Vázquez et al. (2020) consists in using fixed-size attention bridge (FSAB) designs. FSAB models, noted $\mathcal{L}$, resemble $\mathcal{T}$ models except for the fact that the fully-shared Transformer layer bridge is replaced by the structured embedding architecture proposed by Lin et al. (2017):

$$\mathbf{Y} = \mathrm{softmax}\left(\mathbf{W}_Q \,\mathrm{ReLU}\left(\mathbf{W}_K \mathbf{X}\right)^\top\right) \cdot \mathbf{X} \quad (1)$$

with $\mathbf{X}$ the input matrix of the shared layer. Models of the $\mathcal{L}$ architecture contain language-specific encoders comprising 5 Transformer layers, followed by one FSAB layer shared across all languages.

3.2 Datasets

We use two MT datasets: the United Nations Parallel Corpus (Ziemski et al., 2016, UNPC), which contains documents in six UN languages (Arabic, Chinese, English, French, Russian, and Spanish); and OPUS100 (Zhang et al., 2020), an English-centric multilingual corpus derived from Tiedemann (2012) spanning 100 languages. We ignore all OPUS translation directions not present in UNPC. Since the UNPC contains over 10M paired sentences across six languages (Arabic, English, Spanish, French, Russian, Mandarin Chinese), we consider the entire released data, rather than the fully aligned sub-corpus, and hold out 10% of the data for any evaluation and/or experiments. We ensure that sentences are unique to a split, i.e., if a pair of sentences (s_1, s_2) is present in the test split, then any pair (s_1, s_3) involving either of these sentence will also be assigned to the test split. Out of these 10%, we randomly select 25k sentences per language pairs to use as test sets. The remaining 90% examples are used for training, with 10k sentences per language pairs set aside for validation.

Test splits for generalization. We assess generalization capabilities in two common setups: zero-shot translation directions and out-of-distribution

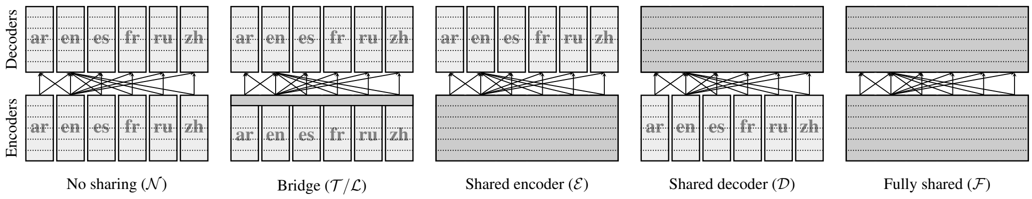


Figure 1: Overview of considered architectures, focusing on **EN** setting (using English as a pivot). Layers shaded in dark gray are shared across all languages; layers shaded in light gray are specific to a source or target language.

(OOD) examples. To evaluate out-of-distribution performances, we simply train models on one dataset (UNPC or OPUS) and evaluate it on the other (resp. OPUS or UNPC). Since bridge components are argued to be useful for unseen translation directions (Liao et al., 2021), we experiment with different language pivots to artificially create zero-shot translation directions. We construct three distinct UNPC training sets: (i) one using all 30 translation directions available in the UNPC, ("**All**"); (ii) one using all 10 directions involving English as a source or target ("**EN**"); and (iii) one using all 10 directions involving Arabic as a source or target ("**AR**"). This allows us to evaluate our models in both English-centric and non-English-centric contexts as well as in a zero-shot setting.[2] Hence, we refer to **EN** or **AR** being pivot languages, when an experiment is centered around that language.

Training conditions. To enable zero-shot translation (Vázquez et al., 2019; Artetxe and Schwenk, 2019, cf.), we train our models on auto-encoding tasks for all 6 languages. UNPC models are trained on monolingual data derived from the UNPC, and likewise OPUS models are trained on OPUS monolingual data. We train three seeds of all six model variants ($\mathcal{F}$, $\mathcal{N}$, $\mathcal{E}$, $\mathcal{D}$, $\mathcal{T}$, $\mathcal{L}$) on the four training sets (**UNPC-All**, **UNPC-EN UNPC-AR**, **OPUS-EN**) under a strictly controlled computational budget: All models are exposed to the same number of datapoints and are trained with 6 AMD MI250X GPUs. We use the hyperparameters of Boggia et al. (2023) aside from batch accumulation, set to 8. We use $k = 50$ in $\mathcal{L}$ models as Vázquez et al. (2020).

4 Results

The primary metric used for evaluating the performance of our models is BLEU (Papineni et al., 2002; Post, 2018).[3] Results are shown in Table 1.

Choice of architecture. A clear trend emerges from our results: Across the board, the encoder-shared models $\mathcal{E}$ are found to be the most successful, followed by the fully-shared, non-modular models $\mathcal{F}$. The latter only prevails upon the former in Arabic-centric scenario. At times, these architectures outrank other models considered by large margins of up to 7.5 BLEU points. While fully modular $\mathcal{N}$ models or FSAB-based $\mathcal{L}$ models perform well in the **EN**-centric scenario, these are not overwhelmingly better than $\mathcal{F}$.

Choice of pivot language. We experiment with different pivot languages, **EN** and **AR**, to understand their influence on the results. Our observations indicate that the choice of a pivot language can significantly impact the outcomes: The results with **AR** are always below the corresponding scores with **EN** on translation directions studied during training, whereas **AR** models yield generally higher performance in zero-shot conditions than their **EN** counterparts. Furthermore, we find tentative evidence that the behavior in **EN** and **AR** differs from that of **All**: In the latter case, we find a more limited impact of the architecture being used, with score varying at most by ± 4.2 BLEU points; whereas we observe a spread of up to ± 7.3 BLEU points for the former. As one would expect, being exposed to all translation directions during training (**All**) allows to improve performances averaged all translation directions. If we restrict ourselves to directions a model was exposed to during training, we find that **EN** models often outperform **All** models; whereas **AR** models are more in line with the values we see for **All**. This would suggest that there is a difficulty inherent to the translation directions considered; focusing only on directions that involve English may inflate performances.

Translation directions (seen vs. unseen). Expanding on what we already briefly touched on, we systematically find performances in zero-shot

[2]Since OPUS100 is English-centric, only one variant of this dataset is considered for training.

[3]While COMET (Rei et al., 2020) would in principle be preferable, computing it for all translation directions in every model in our study is prohibitively costly.

Test	Train	Translation directions		$\mathcal{N}$	$\mathcal{F}$	$\mathcal{E}$	$\mathcal{D}$	$\mathcal{T}$	$\mathcal{L}$
test on **UNPC**		**All**	(seen)	26.6 ± 0.5	$\mathbf{28.2 \pm 1.1}$	$\underline{\mathbf{29.0 \pm 0.2}}$	24.8 ± 0.2	26.6 ± 0.1	26.4 ± 0.2
	train on **UNPC**	**AR**	all	18.1 ± 0.3	$\underline{\mathbf{24.6 \pm 1.3}}$	$\mathbf{23.1 \pm 1.7}$	15.7 ± 1.8	16.8 ± 0.4	17.9 ± 0.1
		AR	seen	26.4 ± 0.1	$\underline{\mathbf{27.1 \pm 0.9}}$	$\mathbf{26.6 \pm 0.7}$	22.0 ± 1.0	24.9 ± 0.1	26.2 ± 0.0
		AR	unseen	13.9 ± 0.3	$\underline{\mathbf{23.4 \pm 1.4}}$	$\mathbf{21.4 \pm 2.3}$	12.5 ± 2.2	12.8 ± 0.6	13.7 ± 0.1
		EN	all	19.3 ± 0.4	$\mathbf{22.8 \pm 2.9}$	$\underline{\mathbf{23.9 \pm 1.0}}$	17.9 ± 0.4	19.3 ± 0.1	19.4 ± 0.1
		EN	seen	34.5 ± 0.2	33.1 ± 2.6	$\underline{\mathbf{35.9 \pm 0.1}}$	31.6 ± 0.9	34.0 ± 0.2	$\mathbf{34.6 \pm 0.3}$
		EN	unseen	11.7 ± 0.6	$\mathbf{17.6 \pm 3.0}$	$\underline{\mathbf{17.9 \pm 1.4}}$	11.0 ± 1.0	11.9 ± 0.1	11.8 ± 0.1
	train on **OPUS**	**EN**	all	16.4 ± 0.2	$\mathbf{20.6 \pm 0.5}$	$\underline{\mathbf{20.9 \pm 0.4}}$	13.6 ± 1.2	16.8 ± 0.2	16.3 ± 0.1
		EN	seen	$\mathbf{30.8 \pm 0.2}$	30.5 ± 0.5	$\underline{\mathbf{31.1 \pm 0.3}}$	23.7 ± 1.5	30.7 ± 0.3	30.6 ± 0.3
		EN	unseen	9.1 ± 0.3	$\mathbf{15.6 \pm 0.5}$	$\underline{\mathbf{15.8 \pm 0.5}}$	8.5 ± 1.0	9.9 ± 0.1	9.1 ± 0.1
test on **OPUS**		**All**	(seen)	17.6 ± 0.2	$\mathbf{19.1 \pm 0.8}$	$\underline{\mathbf{19.7 \pm 0.2}}$	16.3 ± 0.2	17.5 ± 0.2	17.5 ± 0.3
	train on **UNPC**	**AR**	all	12.3 ± 0.2	$\underline{\mathbf{16.5 \pm 0.8}}$	$\mathbf{15.4 \pm 1.0}$	10.3 ± 1.4	11.4 ± 0.2	12.1 ± 0.1
		AR	seen	17.7 ± 0.1	$\underline{\mathbf{18.4 \pm 0.8}}$	$\mathbf{17.9 \pm 0.6}$	13.8 ± 1.0	16.6 ± 0.1	17.6 ± 0.1
		AR	unseen	9.2 ± 0.3	$\underline{\mathbf{15.5 \pm 0.8}}$	$\mathbf{13.9 \pm 1.3}$	8.4 ± 1.6	8.5 ± 0.2	9.0 ± 0.1
		EN	all	13.4 ± 0.3	$\mathbf{15.9 \pm 2.0}$	$\underline{\mathbf{17.0 \pm 0.5}}$	12.6 ± 0.2	13.3 ± 0.1	13.5 ± 0.2
		EN	seen	19.7 ± 0.1	$\mathbf{19.8 \pm 1.4}$	$\underline{\mathbf{21.0 \pm 0.0}}$	18.5 ± 0.7	19.2 ± 0.1	$\mathbf{19.8 \pm 0.2}$
		EN	unseen	8.2 ± 0.3	$\mathbf{12.7 \pm 2.5}$	$\underline{\mathbf{13.6 \pm 0.9}}$	7.7 ± 0.7	8.4 ± 0.3	8.2 ± 0.3
	train on **OPUS**	**EN**	all	15.0 ± 0.2	$\mathbf{17.8 \pm 0.3}$	$\underline{\mathbf{17.9 \pm 0.3}}$	12.4 ± 1.0	15.5 ± 0.2	14.8 ± 0.1
		EN	seen	$\underline{\mathbf{25.1 \pm 0.2}}$	24.6 ± 0.3	$\underline{\mathbf{25.1 \pm 0.2}}$	19.7 ± 1.6	24.9 ± 0.2	24.9 ± 0.3
		EN	unseen	6.6 ± 0.2	$\underline{\mathbf{12.1 \pm 0.3}}$	$\mathbf{11.8 \pm 0.5}$	6.3 ± 0.7	7.7 ± 0.2	6.4 ± 0.1

Table 1: Summary of performances, with <u>**best**</u> and **second best** values highlighted (avg. of 3 seeds $\pm$ std. dev.), and broken down according to whether the translation direction was seen during training or not (i.e., zero shot).

conditions to remain firmly below what we observe for translation directions observed during training. This holds across pivot languages and architectures. We do not observe that bridges ($\mathcal{T}$ or $\mathcal{L}$) provide benefits in terms of zero-shot performances over fully modular systems ($\mathcal{N}$). instead, it would appear that sharing the encoder ($\mathcal{E}$ or $\mathcal{F}$) is beneficial—although it is uncertain that this is due to greater generalization capabilities rather than overall improved performances, the improvement brought about by $\mathcal{E}$ and $\mathcal{F}$ models is more substantiated in zero-shot settings (with a gap of at least 4.1 BLEU points in zero-shot settings, whereas $\mathcal{F}$ can be outperformed by $\mathcal{L}$ and/or $\mathcal{N}$ for training directions).

In-distribution vs. out-of-distribution. Comparing performances in-distribution and out-of-distribution does not suggest that bridges meaningfully improve generalization capabilities. Performances of $\mathcal{T}$ and $\mathcal{L}$ models are in line with what we observe for the bridge-less $\mathcal{N}$ models.

5 Statistical modeling

SHAP analysis & predictors importance Are our observations statistically significant? To establish which factors are at play, we rely on SHAP (Lundberg and Lee, 2017), a library and algorithm to derive heuristics for Shapley values (Shapley, 1953). We fit a gradient boosting decision tree

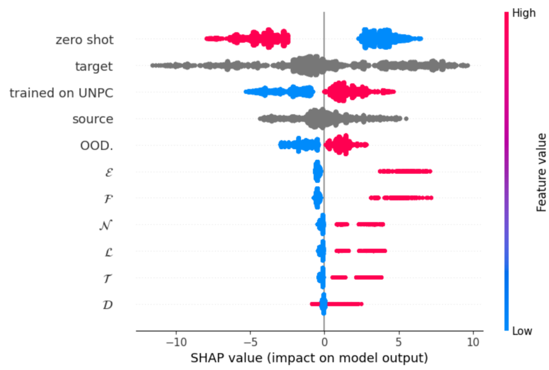


Figure 2: Overview of SHAP values, sorted by mean absolute value. Grey: categorical predictors; red: binary predictors where the value is true; blue, where it is false.

regression model with CatBoost (Prokhorenkova et al., 2018) to explain the BLEU scores obtained on specific language pairs and datasets by all the models we trained. We use as predictors (i) the source language (categorical); (ii) the target language (categorical), (iii) whether the model was trained on UNPC (binary); (iv) whether this translation direction in zero-shot (binary); (v) whether this test corresponds to an out-of-distribution setting (binary); as well as (vi–xi) which architecture is used (binary predicates for each of $\mathcal{N}$, $\mathcal{F}$, $\mathcal{E}$, $\mathcal{D}$, $\mathcal{T}$, and $\mathcal{L}$).

Figure 2 provides a general overview of the results of this analysis. The exact evaluation

	coef	std err	t	P> |t|
Intercept	11.8312	0.211	56.153	0.000
has bridge	4.4287	0.248	17.851	0.000
shares enc	5.3919	0.248	21.733	0.000
zero shot	−7.5567	0.139	−54.277	0.000
OOD	1.9472	0.140	13.917	0.000
from EN	1.9792	0.178	11.099	0.000
from ES	2.7001	0.204	13.235	0.000
from FR	1.5625	0.182	8.578	0.000
from RU	0.5932	0.182	3.257	0.001
from ZH	−2.6625	0.182	−14.617	0.000
to EN	9.1120	0.178	51.098	0.000
to ES	7.7651	0.204	38.061	0.000
to FR	4.0147	0.182	22.040	0.000
to RU	2.0937	0.182	11.494	0.000
to ZH	−5.2907	0.182	−29.046	0.000
trained on UNPC	4.3278	0.125	34.705	0.000
has bridge×zero shot	−4.7733	0.283	−16.876	0.000
has bridge×OOD	0.4169	0.283	1.472	0.141
shares enc×zero shot	0.3607	0.283	1.275	0.202
shares enc×OOD	0.9785	0.283	3.455	0.001

Table 2: OLS coefficients and significance. Intercept: $\mathcal{N}$-type, not OOD, not zero-shot, from & to AR.

conditions—i.e. the training and testing corpora and the specific language pairs seen at training and during the test at hand all, corresponding to predictors (i–v)—have a strong impact on the observed BLEU scores. We also see that using models of type $\mathcal{F}$ and $\mathcal{E}$ more strongly and more positively impacts the BLEU scores we observe than any other model type. In short, we find that most modular models fail to bring about results comparable with what we see for our non-modular baseline $\mathcal{F}$, with the sole exception of encoder-sharing $\mathcal{E}$.

OLS model & predictors interaction. Is there evidence that some modular architectures (and bridges in particular) enhance generalization capabilities? While SHAP values provide independent coefficients for each factor, this question is at its core one of interrelation—and is thus best studied through models able to capture potential interactions between predictors. To that end, we fit a simple ordinary least squares (OLS) linear model to predict the BLEU scores of our models using as predictors (i) whether the architecture contains a bridge (i.e., models of type $\mathcal{T}$ or $\mathcal{L}$); (ii) whether it shares the encoder across source languages (i.e., models of type $\mathcal{F}$ or $\mathcal{E}$); (iii) whether the model is tested in zero-shot; (iv) whether it is tested in an OOD setting; (v) whether the model was trained on UNPC; (vi & vii) the source and target languages; (viii–xi) the interactions between modular design (i.e., predictors i & ii) and performances in generalization conditions (viz. predictors iii & iv).[4]

Our model achieves a R^2 of 0.763. Predictor coefficients and significance are listed in Table 2. As expected, modular design and training & test conditions (predictors i–vii) are always significant. Zero shot performances are linked to the strongest negative coefficient in our model; likewise, translating from or to ZH also turns out to degrade performance somewhat compared to the intercept (AR). Looking at interactions, we find that models with a bridge require a clear *negative* correction in zero-shot scenarios, *opposite* to what has been argued by Liao et al. (2021). Models of type $\mathcal{F}$ and $\mathcal{E}$ require a positive correction in OOD settings, suggesting they distinguish themselves further from other modular architectures. This statistical modeling suggests that bridge-based architectures significantly decrease generalization capabilities, as opposed to other modular ($\mathcal{E}$) and non-modular ($\mathcal{F}$) designs—in contrast with much of the discourse about their benefit for language independence and usefulness in zero-shot conditions (Raganato et al., 2019; Zhu et al., 2020; Vázquez et al., 2020).

6 Conclusions

In this work, we study the claim that bridge layers in modular architectures foster greater generalization capabilities. Given a carefully controlled computational budget, bridge architectures never clearly outperform bridge-less architectures, be they modular or not. In particular, we find non-modular architectures exhibit strong competitiveness, as they are only outperformed by modular architectures with language independent encoders and modular language-specific decoders. Additionally, we note that training conditions, such as the translation direction accessible to a model during training, have a significant impact.

These results suggest that current modular architectures, especially those using bridging layers, have limited potential insofar MT is concerned. In most cases, a default non-modular transformer fares better or just as well than the most effective modular system. Our study focused on modular architectures in a small-scale, well controlled experimental protocol; we leave questions such as whether these remarks carry on at a larger scale, both of model parameter counts and number of languages concerned, for future work.

[4]We ignore datapoints from type $\mathcal{D}$ models since we are not aware of specific claims with respect to this architecture.

Acknowledgements

This work is part of the FoTran project, funded by the European Research Council (ERC) under the EU's Horizon 2020 research and innovation program (agreement № 771113). We also thank the CSC-IT Center for Science Ltd., for computational resources and NVIDIA AI Technology Center (NVAITC) for the expertise in distributed training.

References

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*, 7:597–610.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Michele Boggia, Stig-Arne Grönroos, Niki Loppi, Timothee Mickus, Alessandro Raganato, Jörg Tiedemann, and Raúl Vázquez. 2023. Dozens of translation directions or millions of shared parameters? comparing two types of multilinguality in modular machine translation. In *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 238–247, Tórshavn, Faroe Islands. University of Tartu Library.

Carlos Escolano, Marta R. Costa-jussà, José A. R. Fonollosa, and Mikel Artetxe. 2021. Multilingual machine translation: Closing the gap between shared and language-specific encoder-decoders. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 944–948, Online. 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.

Junwei Liao, Yu Shi, Ming Gong, Linjun Shou, Hong Qu, and Michael Zeng. 2021. Improving zero-shot neural machine translation on language-specific encoders- decoders. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *International Conference on Learning Representations*.

Yichao Lu, Phillip Keung, Faisal Ladhak, Vikas Bhardwaj, Shaonan Zhang, and Jason Sun. 2018. A neural interlingua for multilingual machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 84–92, Brussels, Belgium. Association for Computational Linguistics.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Zhuoyuan Mao, Haiyue Song, Raj Dabre, Chenhui Chu, and Sadao Kurohashi. 2023. Variable-length neural interlingua representations for zero-shot neural machine translation.

Timothee Mickus, Stig-Arne Grönroos, Joseph Attieh, Michele Boggia, Ona De Gibert, Shaoxiong Ji, Niki Andreas Loppi, Alessandro Raganato, Raúl Vázquez, and Jörg Tiedemann. 2024. MAMMOTH: Massively multilingual modular open translation @ Helsinki. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 127–136, St. Julians, Malta. 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.

Telmo Pessoa Pires, Robin M. Schmidt, Yi-Hsiu Liao, and Stephan Peitz. 2023. Learning language-specific layers for multilingual machine translation.

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.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. Catboost: unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Taido Purason and Andre Tättar. 2022. Multilingual neural machine translation with the right amount of sharing. In *Proceedings of the 23rd Annual Conference of the European Association for Machine Translation*, pages 91–100, Ghent, Belgium. European Association for Machine Translation.

Alessandro Raganato, Raúl Vázquez, Mathias Creutz, and Jörg Tiedemann. 2019. An evaluation of language-agnostic inner-attention-based representations in machine translation. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 27–32, Florence, Italy. 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.

Richard H. Richens. 1956. Preprogramming for mechanical translation. *Mechanical Translation*, 3(1):20–25.

Lloyd S Shapley. 1953. A value for n-person games. In Harold W. Kuhn and Albert W. Tucker, editors, *Contributions to the Theory of Games II*, pages 307–317. Princeton University Press, Princeton.

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, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Raúl Vázquez, Alessandro Raganato, Mathias Creutz, and Jörg Tiedemann. 2020. A systematic study of inner-attention-based sentence representations in multilingual neural machine translation. *Computational Linguistics*, 46(2):387–424.

Raúl Vázquez, Alessandro Raganato, Jörg Tiedemann, and Mathias Creutz. 2019. Multilingual NMT with a language-independent attention bridge. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 33–39, Florence, Italy. Association for Computational Linguistics.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

Changfeng Zhu, Heng Yu, Shanbo Cheng, and Weihua Luo. 2020. Language-aware interlingua for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1650–1655, Online. Association for Computational Linguistics.

Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The United Nations parallel corpus v1.0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3530–3534, Portorož, Slovenia. European Language Resources Association (ELRA).

Knowledge Distillation vs. Pretraining from Scratch under a Fixed (Computation) Budget

Minh Duc Bui[▽] **Fabian David Schmidt**[◇]
Goran Glavaš[◇] **Katharina von der Wense**[▽♠]
[▽]Johannes Gutenberg University Mainz, Germany
[◇]Center For Artificial Intelligence and Data Science, University of Würzburg, Germany
[♠]University of Colorado Boulder, USA
{minhducbui, k.vonderwense}@uni-mainz.de
{fabian.schmidt, goran.glavas}@uni-wuerzburg.de

Abstract

Compared to standard language model (LM) pretraining (i.e., from scratch), Knowledge Distillation (KD) entails an additional forward pass through a teacher model that is typically substantially larger than the target student model. As such, KD in LM pretraining materially slows down throughput of pretraining instances vis-a-vis pretraining from scratch. Scaling laws of LM pretraining suggest that smaller models can close the gap to larger counterparts if trained on more data (i.e., processing more tokens)—and under a fixed computation budget, smaller models *are able* be process more data than larger models. We thus *hypothesize that* KD *might, in fact, be suboptimal to pretraining from scratch for obtaining smaller LMs, when appropriately accounting for the compute budget*. To test this, we compare pretraining from scratch against several KD strategies for masked language modeling (MLM) in a *fair* experimental setup, with respect to amount of computation as well as pretraining data. Downstream results on GLUE, however, do *not* confirm our hypothesis: while pretraining from scratch performs comparably to ordinary KD under a fixed computation budget, more sophisticated KD strategies, namely TinyBERT (Jiao et al., 2020) and MiniLM (Wang et al., 2023), outperform it by a notable margin. We further find that KD yields larger gains over pretraining from scratch when the data must be repeated under the fixed computation budget.[1]

1 Introduction

Knowledge distillation (KD; Hinton et al., 2015; Jiao et al., 2020) during LM pretraining has emerged as the primary mean of compressing the capabilities of a large pretrained teacher model into a task agnostic smaller student model. KD is praised for yielding high-performing task agnostic small models, mitigating the need for pretraining

[1]Code is available at https://github.com/MinhDucBui/revisiting_distillation.

Name	Identical Architect.	Identical Compute
DistilBERT (Sanh et al., 2020)	***No***	***No***
TinyBERT (Jiao et al., 2020)	***Yes***	***No***
MobileBERT (Sun et al., 2020)	***No***	***No***
MiniLM (Wang et al., 2020)	***No***	***No***
Our Work	***Yes***	***Yes***

Table 1: Assessing the fairness of evaluation setups in previous works for task-agnostic masked language models, trained with KD and without KD.

(small models) from scratch, which is typically considered more expensive. The body of existing KD work for MLM (Jiao et al., 2020; Wang et al., 2023), however, typically does not compare KD against pretraining from scratch in a *fair setup*: (i) with the same target models (exactly the same architecture) and (ii) under the same computation budget. Compared to just training the target model from scratch, KD comes with a computational overhead of forward passes through the typically considerably larger teacher model. This, under the same computation budget, allows pretraining from scratch to consume more data (i.e., more tokens) than KD, which leads to the central research question of this work: *in a fair setup where both are given equal overall computation budget, is* KD *still more effective than pretraining from scratch (*No-KD*)?* We hypothesize that, under a fair evaluation setup, No-KD may be as effective as KD, rendering KD inconsequential. Our reasoning is based on two observations:

1) Fair KD Comparison. A fair comparison, in which both setups are given identical computation budgets (as well as identical target models) eludes existing work on KD. Jiao et al. (2020) compare their model to $\text{BERT}_{\text{Tiny}}$ (Turc et al., 2019), which has the same architecture but employs significantly different training resources than their $\text{TinyBERT}_{\text{Tiny}}$, preventing a fair comparison. Similarly, Sanh et al. (2020) compare their distilled stu-

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 41–47
June 20, 2024 ©2024 Association for Computational Linguistics

dent solely against the teacher, whereas Sun et al. (2020); Wang et al. (2020) only add comparison against larger pretrained models and competing KD strategies. Even the body of work that focuses on comparing different KD strategies has only recently sought to standardize training and thus enable fair comparisons (Lu et al., 2022; Wang et al., 2023).

2) Scaling Laws. Scaling laws (Kaplan et al., 2020; Hoffmann et al., 2022), reveal that, under a fixed computation budget, only a marginal correlation exists between the LM size and it's performance: smaller models compensate their lower learning efficiency with the ability to process more tokens within the same budget. While there are ongoing refinements to the scaling law (Hoffmann et al., 2022), it has been consistently reaffirmed by several studies (Geiping and Goldstein, 2023; Bansal et al., 2022; Clark et al., 2022). For instance, Geiping and Goldstein (2023) showcases this behavior by training multiple BERT models with varying architecture sizes for a fix 24 hour duration, resulting in similar loss values across all sizes.

Contribution Motivated by the recent findings in the realm of scaling laws and recognizing the absence of a fair comparison between KD and No-KD, our primary contribution lies in the comparison of No-KD against KD strategies for MLM while ensuring a fair setup with regards to compute budget and pretraining data. We initially assess No-KD in an optimal setup, where unlimited pretraining tokens are available within a fixed compute budget. Additionally, we examine the scenario when data is constrained within a fixed compute budget.

Our results reveal that, in the optimal setting, No-KD performs indeed comparably to vanilla-KD, exhibiting an average improvement over vanilla-KD of 0.4 and 0.1 points for 6-, and 12-layer models on GLUE. However, No-KD falls short of surpassing more advanced KD strategies, exemplified by the comparison with TinyBERT and MiniLM. When available data is limited within the fixed compute budget, KD strategies outperform No-KD by an even larger margin: No-KD, though faster, needs more epochs, whereas KD strategies extract more information from limited data.

2 Distillation Strategies

Vanilla-KD Vanilla-KD for MLM pretraining is set up as follows. A small MLM student is trained to mimick the predictions for a particular training instance of a large pretrained MLM teacher: The distillation objective is to minimize the soft cross-entropy between the logits $\mathbf{z}^T$ of the MLM teacher and the logits $\mathbf{z}^S$ of the MLM student, with a temperature factor t: $\mathcal{L}_{\text{pred}} = \texttt{CE}(\mathbf{z}^T/t, \mathbf{z}^S/t)$. Following Hinton et al. (2015), the final training loss equally combines $\mathcal{L}_{\text{pred}}$ with the MLM loss $\mathcal{L}_{\text{CE}}$ during pretraining.

TinyBERT Jiao et al. (2020) distill knowledge by minimizing the mean-squared error (MSE) between latent representations of the MLM student S and the MLM teacher T by model layers as follows. First, the embedding matrices of the student ($\mathbf{E_S}$) and the teacher ($\mathbf{E_T}$) are aligned by minimizing the loss $\mathcal{L}_{\text{embd}} = \texttt{MSE}(\mathbf{E}^S\mathbf{W}_e, \mathbf{E}^T)$. The authors further fit the unnormalized attention scores per head h of the MLM student S to the MLM teacher T by optimizing $\mathcal{L}_{\text{att}} = 1/h \sum_{i=1}^{h} \texttt{MSE}(\mathbf{A}_i^S, \mathbf{A}_i^T)$. Lastly, the output hidden states $\mathbf{H}^S$ of transformer layers of the student are also regressed onto the corresponding teacher output representations $\mathbf{H}^T$ by optimizing $\mathcal{L}_{\text{hid}} = \texttt{MSE}(\mathbf{H}^S\mathbf{W_h}, \mathbf{H}^T)$.[2]

MiniLM Wang et al. (2020) also mimic the self-attention modules of the MLM teacher. Unlike TinyBert, MiniLM focuses on the last attention module. Wang et al. (2020) minimize the KL-divergence between the self-attention distributions of the MLM teacher and the MLM student. They further minimize the KL-divergence between the value relations of the MLM teacher T and MLM student S, i.e. $\mathcal{L}_{VR} = \frac{1}{A_h|x|}\sum_{a=1}^{A_h}\sum_{t=1}^{|x|} D_{KL}(\mathbf{VR}^T||\mathbf{VR}^S)$. The value-relation denotes the outer product of values $\mathbf{V}$ across heads in the last attention module, i.e. $\mathbf{VR} = softmax(\frac{\mathbf{VV}^\intercal}{\sqrt{d}})$.

3 Experiment Setup

Model Architectures We experiment with two different teacher and student sizes: First, we use a 12-layer pretrained $\text{BERT}_{\text{base}}$ (Devlin et al., 2019) model (L=12, H=768, A=12, Total Parameters=110M) as the teacher and a randomly initialized 6-layer BERT_6 model (L=6, H=768, A=12, Total Parameters=67M) as the student. We then scale the setting up to a pretrained $\text{BERT}_{\text{Large}}$ (L=24,

[2]The distillation of embeddings $\mathbf{E}$ and output hidden states $\mathbf{H}$ is learned up to projection matrices $\mathbf{W}_{e;h}$ matrices to bridge varying dimensionalities of representations across architectures.

	Total Token Throughput	QNLI (Acc)	SST-2 (Acc)	MNLI (Acc)	MRPC (F1)	QQP (Acc)	RTE (Acc)	CoLA (Mcc)	Avg	Δ
6-Layer: Unlimited Pretraining Tokens within Fixed Compute Budget										
No-KD$_6$	4.6B	86.5	89.8	79.2	87.3	90.1	60.7	**47.3**	77.3	–
Vanilla-KD$_6$	2.6B	87.3	89.2	78.8	87.1	89.7	61.7	44.8	76.9	−0.4
MiniLM$_6$	2.6B	88.5	90.6	81.7	90.0	90.3	64.3	43.9	78.5	+1.2
TinyBERT$_6$	2.6B	**89.5**	**91.0**	**82.2**	**90.3**	**90.4**	**67.2**	40.8	**78.8**	+1.5
12-Layer: Unlimited Pretraining Tokens within Fixed Compute Budget										
No-KD$_{12}$	4.6B	87.8	90.1	80.6	86.5	90.3	60.3	**51.1**	78.1	–
Vanilla-KD$_{12}$	2.6B	86.3	90.5	79.8	88.8	89.9	62.1	48.6	78.0	−0.1
MiniLM$_{12}$	2.6B	**90.0**	91.2	**83.3**	90.1	**90.9**	**69.0**	49.1	**80.5**	+1.4
TinyBERT$_{12}$	2.6B	89.5	**91.4**	82.0	**90.8**	90.6	65.5	41.1	78.7	+0.6
6-Layer: Limited Pretraining Tokens within Fixed (Increased) Compute Budget										
No-KD$_6$	27.9B	88.8	91.2	81.3	88.0	90.4	59.6	50.5	78.5	–
Vanilla-KD$_6$	15.4B	86.9	91.1	81.1	89.5	90.3	61.7	**58.3**	79.8	+1.3
MiniLM$_6$	15.6B	90.0	91.5	83.0	**90.3**	90.6	65.7	50.7	80.3	+1.8
TinyBERT$_6$	15.6B	**90.5**	**92.3**	**83.3**	90.2	**90.8**	**67.5**	51.8	**80.9**	+2.4

Table 2: Upper part: optimal scenario for No-KD – unlimited pretraining tokens within a fixed compute budget. Lower part: limited data within a fixed compute budget. We present the performance results on the GLUE development set, maintaining a consistent pretraining wall-clock time across all models within each group. The column **Avg** represents the average performance across all tasks, while **Δ** quantifies the average difference between No-KD$_{xx}$ and the other distillation strategies.

H=1024, A=16, Total Parameters=340M) teacher and a randomly initialized 12-layer BERT$_{12}$ student. To speed up the training pipeline and convergence, we use the implementation of Izsak et al. (2021) for the models.

Data We follow BERT (Devlin et al., 2019) and pretrain all models on the Toronto BooksCorpus (Zhu et al., 2015) and English Wikipedia.[3] After MLM pretraining, we finetune and evaluate the models on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018), a collection of diverse natural language understanding tasks.[4]

Pretraining We first pretrain a BERT$_6$ model from scratch (without KD). This model is denoted as No-KD$_6$. We further apply the KD strategies (cf. §2), for which BERT$_{base}$ (BERT$_{large}$) is the MLM teacher for the 6-layer (12-layer) MLM student. The resulting 6-layer models are indicated with a subscript 6, while the 12-layer models are marked with a subscript 12. Notably, we only employ the last layer for layer-wise distillation, as we confirm the findings of Wang et al. (2023) that distilling knowledge of multiple layers does not yield consistent performance improvements. We refer to Appendix A.2 for additional hyperparameter, hardware details and wall-clock training time.

Downstream Finetuning We perform a grid search over batch sizes {16, 32} and learning rates {1e-5, 3e-5, 5e-5, 8e-5} to identify the ideal hyperparameters for each task on the GLUE benchmark. We train all configurations for 5 epochs. We utilize a polynomial learning rate schedule and a maximum sequence of 128.

4 Results

4.1 Setting: Unlimited Data with Fixed Compute

We assess No-KD for a single epoch and fix the resulting training wall-clock time for the distillation strategies. Within this compute budget, we train on unlimited pretraining tokens without the need for sample repetition. We report our main results in the upper segment of Table 2.

Low KD Token Throughput We find that the token throughput of No-KD$_6$ and No-KD$_{12}$ is approximately 1.8 times greater than that of the distillation models. This observation underscores that the presence of a teacher model greatly reduces the speed of pretraining.

[3] In November 2023, we crawled the English Wikipedia using Attardi (2015). The official BookCorpus is no longer accessible; however, it was re-crawled by Kobayashi (2018).

[4] We refer to Appendix A.3 for more information about the GLUE datasets.

Performance of 6-layer Students We observe that No-KD$_6$ surpasses Vanilla-KD$_6$ by an average of 0.4 points. This result indicates that Vanilla-KD$_6$ does not exceed pretraining from scratch in a fair setting. However, more advanced KD strategies exhibit notable performance gains over No-KD. On average, TinyBERT$_6$ outperforms No-KD$_6$ by 1.5 points, while MiniLM$_6$ achieves a 1.2 point advantage. These findings suggest that pretraining from scratch falls short in outperforming sophisticated distillation strategies in a fair setup, even when exposed to a higher volume of tokens. The only exception to this trend is CoLA (Corpus of Linguistic Acceptability) (Warstadt et al., 2019), on which No-KD$_6$ excels.

Performance of 12-layer Students We find the same pattern when we double the number of transformer layers in MLM students: Vanilla-KD$_{12}$ fails to outperform No-KD$_{12}$, yet it is surpassed by MiniLM$_{12}$ by an average of 1.4 points. Notably, No-KD$_{12}$ once again exhibits superior performance on the CoLA task compared to other strategies.

CoLA Performance No-KD$_6$ demonstrates superior performance on CoLA, surpassing the next most effective strategy by 2.5 and 2.0 points for 6- and 12-layer models, respectively. We hypothesize that CoLA benefits significantly from masked language modelling, as evidenced by the improved performance of Vanilla-KD on CoLA compared to other distillation strategies, aligning with findings by Wang et al. (2023). Another contributing factor could be the scalability of CoLA with respect to tokens encountered during pretraining. This observation contradicts the results of Liu et al. (2021), who suggest that CoLA can be learned relatively quickly compared to other downstream tasks. However, it aligns with the conclusions of Geiping and Goldstein (2023), who also note that their BERT version, exposed to less data, exhibits subpar performance on CoLA.

4.2 Setting: *Limited* Data with Fixed Compute

To extend our findings, we increase the compute budget while retaining a fixed dataset size. We evaluate this setup with 6-layer MLM students and the 12-layer MLM teacher. The analysis provides an estimate of the viability of No-KD when data repetition is necessary within the fixed compute budget. The results are presented in the lower section of Table 2.

The No-KD$_6$ model is underperforming, compared to all distillation strategies, including Vanilla-KD$_6$ by 1.3 points. The performance gap widens even more when compared to MiniLM$_6$ and TinyBERT$_6$, with a substantial difference of 1.8 and 2.4 points on average. We attribute this to the fact that while No-KD benefits from exposure to a larger number of tokens, it also necessitates a larger dataset for effective scaling. Although this requirement can be met in high-resource languages with up-to-date datasets (Kudugunta et al., 2023), it presents a significant challenge in mid to low-resource scenarios. Additionally, No-KD$_6$ is now being outperformed even on CoLA. These results suggest that CoLA's performance indeed needs to process a certain quantity of tokens during pretaining to scale effectively, regardless of additional token repetitions: e.g., the performance of Vanilla-KD$_6$ increases by 13.5 points if scaled from 2.6B unique to 15.4B non-unique pretraining tokens. Interestingly, our findings reveal that Vanilla-KD$_6$ exhibits the best performance on CoLA, underscoring the advantageous impact of masked language modelling on this particular dataset.

5 Discussion

While our study provides insights into a fair evaluation of No-KD and KD for encoder-only models of moderate sizes, revealing negative results for No-KD, it may not cover the full spectrum of model sizes and architectures. For instance, Jha et al. (2023) show that for large decoder-only language models, No-KD performs comparably to Vanilla-KD, aligning with our findings. However, advanced KD strategies like MiniLM exhibit poorer performance than No-KD and Vanilla-KD, challenging both our results and common beliefs about KD regarding large decoder models. This disparity underscores the need for further investigation into a fair KD evaluation across a range of *architectures and scales*. Additionally, we recommend investigating the impact of the teacher budget on performance in the fair setting, a consideration not closely examined in our current work.

6 Conclusion

In this work, we investigate our hypothesis that, provided a fair training scenario, model pretraining from scratch yields similar results as KD during pretraining. Our rationale is grounded in recent advancements in scaling laws for language models and that the literature lacks a fair comparison between No-KD and KD. Our findings demonstrate that our initial assumption does *not* hold true: while, in an optimal setting for No-KD, No-KD performs on par with ordinary KD, it falls short when compared to more sophisticated KD strategies.

Limitations

Firstly, we acknowledge that assessing the compute budget based on training wall-clock time comes with inherent limitations. As outline in Kaddour et al. (2023), wall-clock time can fluctuate even on identical hardware. This fluctuation may arise from factors such as the utilization of non-deterministic operations or hidden background processes. Nevertheless, we only see negligible variations across different runs for the same training pipeline.

Another limitation of our work pertains to data size. Exploring larger pretraining corpora than ours might be worthwhile, although we note that even within our current data scale, KD consistently outperforms No-KD by a significant margin. Even with potential increases in data size, KD remains valuable as it provides a stronger starting point compared to No-KD.

Lastly, we acknowledge that the pretraining corpus is the same as what the teacher used. This shared corpus might influence KD strategies either positively or negatively.

Ethics Statement

We acknowledge our exclusive focus on the English language, overlooking the many challenges of other languages. Additionally, we recognize our sole emphasis on performance metrics, neglecting considerations related to the fairness of the resulting models. We also note that our research extensively employed GPU hours for both pretraining and finetuning, with a keen awareness of the environmental and resource implications associated with such usage.

Acknowledgement

The research in this paper was funded by the Carl Zeiss Foundation, grant number P2021-02-014 (TOPML project).

References

Giuseppe Attardi. 2015. Wikiextractor. https://github.com/attardi/wikiextractor.

Yamini Bansal, Behrooz Ghorbani, Ankush Garg, Biao Zhang, Colin Cherry, Behnam Neyshabur, and Orhan Firat. 2022. Data scaling laws in NMT: The effect of noise and architecture. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 1466–1482. PMLR.

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.

Aidan Clark, Diego De Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, George Bm Van Den Driessche, Eliza Rutherford, Tom Hennigan, Matthew J Johnson, Albin Cassirer, Chris Jones, Elena Buchatskaya, David Budden, Laurent Sifre, Simon Osindero, Oriol Vinyals, Marc'Aurelio Ranzato, Jack Rae, Erich Elsen, Koray Kavukcuoglu, and Karen Simonyan. 2022. Unified scaling laws for routed language models. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 4057–4086. 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, Minneapolis, Minnesota. Association for Computational Linguistics.

Bill Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*. Asia Federation of Natural Language Processing.

Jonas Geiping and Tom Goldstein. 2023. Cramming: Training a language model on a single GPU in one day. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 11117–11143. PMLR.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. 2022. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*, volume 35, pages 30016–30030. Curran Associates, Inc.

Peter Izsak, Moshe Berchansky, and Omer Levy. 2021. How to train BERT with an academic budget. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10644–10652, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ananya Harsh Jha, Tom Sherborne, Evan Pete Walsh, Dirk Groeneveld, Emma Strubell, and Iz Beltagy. 2023. How to train your (compressed) large language model.

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.

Jean Kaddour, Oscar Key, Piotr Nawrot, Pasquale Minervini, and Matt J. Kusner. 2023. No train no gain: Revisiting efficient training algorithms for transformer-based language models.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.

Sosuke Kobayashi. 2018. Homemade bookcorpus. `https://github.com/soskek/bookcorpus`.

Sneha Kudugunta, Isaac Caswell, Biao Zhang, Xavier Garcia, Christopher A. Choquette-Choo, Katherine Lee, Derrick Xin, Aditya Kusupati, Romi Stella, Ankur Bapna, and Orhan Firat. 2023. Madlad-400: A multilingual and document-level large audited dataset.

Zeyu Liu, Yizhong Wang, Jungo Kasai, Hannaneh Hajishirzi, and Noah A. Smith. 2021. Probing across time: What does RoBERTa know and when? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 820–842, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chengqiang Lu, Jianwei Zhang, Yunfei Chu, Zhengyu Chen, Jingren Zhou, Fei Wu, Haiqing Chen, and Hongxia Yang. 2022. Knowledge distillation of transformer-based language models revisited.

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.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

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.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

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, 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.

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 *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Xinpeng Wang, Leonie Weissweiler, Hinrich Schütze, and Barbara Plank. 2023. How to distill your BERT: An empirical study on the impact of weight initialisation and distillation objectives. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1843–1852, Toronto, Canada. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

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.

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

A.1 Implementation Details

We use the code base from Izsak et al. (2021) for both pretraining and finetuning of `No-KD` models. In the case of `KD` models, we utilize the code base introduced by Wang et al. (2023), which itself builds upon the work of Izsak et al. (2021). Our code is available at `https://github.com/MinhDucBui/revisiting_distillation`.

A.2 Pretraining Details

Our pretraining pipeline employs a batch size of 1024, employing gradient accumulation with a batch size of 256. We adopt a time-based learning rate schedule with a linear curve. The peak learning rate is set to 5e-4 for distillation strategies and 1e-3 for `No-KD`. We opt for a warmup proportion of 0.06 for both scenarios. Utilizing the AdamW optimizer with $(\beta_1, \beta_2) = (0.9, 0.98)$ and $\epsilon = 1e - 6$, we conduct training with mixed precision techniques.

We measure compute budget by wall-clock time. All experiments are conducted on NVIDIA A100. Training our 6-layer model for a single epoch requires around 4 hours of wall-clock training time, while the 12-layer model demands approximately 11 hours. Scaling up the 6-layer model to 27.9B tokens extends the training duration to about 24 hours. Fine-tuning on GLUE with a single A100 GPU, coupled with grid-hyperparameter search, consumes up to 50 hours for the 6-layer models and nearly 100 hours for the 12-layer variants.

A.3 GLUE Details

We provide a brief overview of each dataset within GLUE. For additional information regarding each data split, evaluation metric and more, see Wang et al. (2018).

CoLA The Corpus of Linguistic Acceptability (Warstadt et al., 2019) comprises English acceptability judgments sourced from books and journal articles on linguistic theory.

SST-2 The Stanford Sentiment Treebank (Socher et al., 2013) includes sentences extracted from movie reviews, along with human annotations of their binary sentiment.

MRPC The Microsoft Research Paraphrase Corpus (Dolan and Brockett, 2005) consists of sentence pairs automatically extracted from online news sources, with human annotations indicating whether the sentences are semantically equivalent.

QQP The Quora Question Pairs dataset is a compilation of question pairs from the community question-answering website. The objective is to determine whether a pair of questions are semantically equivalent.

STS-B The Semantic Textual Similarity Benchmark (Cer et al., 2017) contains sentence pairs with a human annotated similarity score ranging from 1 to 5.

MNLI The Multi-Genre Natural Language Inference Corpus (Williams et al., 2018) is a crowd-sourced collection of sentence pairs with textual entailment annotations. The task involves predicting whether a premise sentence entails, contradicts, or is neutral with respect to a hypothesis.

QNLI The Stanford Question Answering Dataset (Rajpurkar et al., 2016) is a question-answering dataset comprising question-paragraph pairs, with the task of determining whether the context sentence contains the answer to the question.

RTE The Recognizing Textual Entailment (RTE) datasets originate from annual textual entailment challenges. The dataset is standardized to a two-class split, collapsing neutral and contradiction into "not entailment" for consistency.

An Analysis of BPE Vocabulary Trimming in Neural Machine Translation

Marco Cognetta
Tokyo Institute of Technology
cognetta.marco@gmail.com

Tatsuya Hiraoka
Fujitsu Limited
hiraoka.tatsuya@fujitsu.com

Naoaki Okazaki
Tokyo Institute of Technology
okazaki@c.titech.ac.jp

Rico Sennrich
University of Zurich
sennrich@cl.uzh.ch

Yuval Pinter
Ben-Gurion University of the Negev
uvp@cs.bgu.ac.il

Abstract

We explore threshold vocabulary trimming in Byte-Pair Encoding subword tokenization, a tokenization postprocessing step that replaces rare subwords with their component subwords. The technique is available in popular tokenization libraries but has not been subjected to rigorous scientific scrutiny. While the removal of rare subwords is suggested as best practice in model implementations, both as a means to reduce model size and for improving model performance through robustness, our experiments indicate that, across a large space of hyperparameter settings, vocabulary trimming fails to consistently improve model performance, and is even prone to incurring heavy degradation.

1 Introduction

Subword tokenization is an important process in modern neural language modeling, as it enables models to represent any possible word over a known alphabet while keeping the vocabulary size small. One of the most common subword tokenization methods is Byte-Pair Encoding (BPE; Gage, 1994; Sennrich et al., 2016), a greedy, statistical subword tokenization method. BPE builds its vocabulary and tokenizes a corpus by iteratively replacing the most frequently co-occurring token pair with a single, merged token. An unfortunate side-effect of this process is the existence of "intermediate" subwords—subwords that appear during the process of forming longer subwords, but rarely appear as output tokens in the final sequence.

Vocabulary trimming is a tokenization postprocessing step where subwords that appear fewer than a prescribed number of times in a given corpus are replaced with their component subwords, with the intent of removing rare tokens for which the model cannot learn a robust representation (Sennrich et al., 2017; Sennrich and Zhang, 2019).

Let $\mathcal{B} = (\mathcal{V}_\mathcal{B}, \mathcal{M}_\mathcal{B})$ be a BPE tokenizer trained on corpus $\mathcal{C}$ with character vocabulary Σ. $\mathcal{V}_\mathcal{B} \subset \Sigma^+$ is the subword vocabulary and $\mathcal{M}_\mathcal{B} \subset \mathcal{V}_\mathcal{B} \times \mathcal{V}_\mathcal{B}$ is a set of merges such that $\forall v \in \mathcal{V}_\mathcal{B} \setminus \Sigma$, there exists a unique $(l, r) \in \mathcal{M}_\mathcal{B}$ such that $lr = v$. And, let c_v be the number of times a token v appears in the tokenized corpus and $\mathbb{T} \geq 0$ be a threshold. Then, $\mathcal{X}_{\mathcal{B},\mathbb{T}} = \{v \in \mathcal{V}_\mathcal{B} \setminus \Sigma \mid c_v \leq \mathbb{T}\}$ is the set of non-atomic subword tokens that appear at most $\mathbb{T}$ times in the tokenized corpus and $\mathrm{dec}_{\mathcal{X}_{\mathcal{B},\mathbb{T}}} : \mathcal{V}_\mathcal{B} \to \mathcal{V}_\mathcal{B}^+$ is a recursive decomposition function:

$$\underset{\mathcal{X}_{\mathcal{B},\mathbb{T}}}{\mathrm{dec}}(v) = \begin{cases} v & \text{if } v \notin \mathcal{X}_{\mathcal{B},\mathbb{T}} \\ \mathrm{dec}_{\mathcal{X}_{\mathcal{B},\mathbb{T}}}(l_v) \circ \mathrm{dec}_{\mathcal{X}_{\mathcal{B},\mathbb{T}}}(r_v) & \text{otherwise.} \end{cases}$$

Given a $\mathcal{B}$-tokenized sequence $t_1, t_2, \ldots, t_n$, a trimmed BPE tokenizer produces a new sequence $\mathrm{dec}_{\mathcal{X}_{\mathcal{B},\mathbb{T}}}(t_1), \mathrm{dec}_{\mathcal{X}_{\mathcal{B},\mathbb{T}}}(t_2), \ldots, \mathrm{dec}_{\mathcal{X}_{\mathcal{B},\mathbb{T}}}(t_n)$.

We perform a comprehensive study to understand the actual effect of vocabulary trimming on the performance of machine translation systems. In general, we find that vocabulary trimming has no consistent positive effect on model quality, and in many cases can substantially degrade it.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 48–50
June 20, 2024

Vocabulary $(\mathbb{B}_s, \mathbb{B}_t)$	Thresholds $(\mathbb{T}_s, \mathbb{T}_t)$	BLEU	COMET	Effective Vocabulary $(\hat{\mathbb{B}}_s, \hat{\mathbb{B}}_t)$	Sequence Length
$(6k, 6k)$	Baseline	34.05	79.52	$(6.1k, 6.0k)$	23.30/22.12
	(100, 100)	−0.28	+0.06	$(5.3k, 4.2k)$	+1.4%/+4.2%
	(100, 150)	−0.66	−0.90	$(5.3k, 2.9k)$	+1.4%/+10.6%
	(100, 200)	−0.41	−0.45	$(5.3k, 2.3k)$	+1.4%/+15.6%
	(150, 100)	−0.27	−0.96	$(3.7k, 4.2k)$	+7.5%/+4.2%
	(150, 150)	−0.28	+0.03	$(3.7k, 2.9k)$	+7.5%/+10.6%
	(150, 200)	−0.22	+0.11	$(3.7k, 2.3k)$	+7.5%/+15.6%
	(200, 100)	−0.22	−0.02	$(2.9k, 4.2k)$	+13.1%/+4.2%
	(200, 150)	−0.12	−0.04	$(2.9k, 2.9k)$	+13.1%/+10.6%
	(200, 200)	−0.30	−0.05	$(2.9k, 2.3k)$	+13.1%/+15.6%
$(8k, 8k)$	Baseline	33.63	79.26	$(8.0k, 8.0k)$	22.47/21.51
	(100, 100)	+0.16	+0.54	$(4.8k, 3.7k)$	+7.3%/+9.4%
	(100, 150)	−0.02	+0.38	$(4.8k, 2.6k)$	+7.3%/+16.7%
	(100, 200)	+0.32	+0.35	$(4.8k, 2.1k)$	+7.3%/+22.0%
	(150, 100)	+0.24	+0.39	$(3.3k, 3.7k)$	+14.7%/+9.4%
	(150, 150)	−0.01	+0.20	$(3.3k, 2.6k)$	+14.7%/+16.7%
	(150, 200)	+0.05	+0.11	$(3.3k, 2.1k)$	+14.7%/+22.0%
	(200, 100)	+0.27	+0.31	$(2.6k, 3.7k)$	+21.3%/+9.4%
	(200, 150)	−0.03	+0.13	$(2.6k, 2.6k)$	+21.3%/+16.7%
	(200, 200)	+0.18	+0.30	$(2.6k, 2.1k)$	+21.3%/+22.0%
$(10k, 10k)$	Baseline	33.56	79.20	$(10.0k, 9.9k)$	21.93/21.12
	(100, 100)	+0.37	+0.25	$(4.4k, 3.4k)$	+12.3%/+13.2%
	(100, 150)	+0.30	+0.25	$(4.4k, 2.4k)$	+12.3%/+20.9%
	(100, 200)	+0.14	+0.24	$(4.4k, 1.9k)$	+12.3%/+26.6%
	(150, 100)	+0.14	+0.26	$(3.1k, 3.4k)$	+20.1%/+13.2%
	(150, 150)	+0.23	+0.22	$(3.1k, 2.4k)$	+20.1%/+20.9%
	(150, 200)	+0.24	+0.48	$(3.1k, 1.9k)$	+20.1%/+26.6%
	(200, 100)	+0.31	+0.22	$(2.3k, 3.4k)$	+27.3%/+13.2%
	(200, 150)	+0.18	+0.47	$(2.3k, 2.4k)$	+27.3%/+20.9%
	(200, 200)	+0.18	+0.45	$(2.3k, 1.9k)$	+27.3%/+26.6%

Vocabulary $\mathbb{B}_j$	Thresholds $(\mathbb{T}_s, \mathbb{T}_t)$	BLEU	COMET	Effective Vocabulary $(\hat{\mathbb{B}}_s, \hat{\mathbb{B}}_t)$	Sequence Length
$7k$	Baseline	34.02	79.52	$(6.5k, 4.9k)$	24.11/23.25
	(100, 100)	−0.02	+0.14	$(4.2k, 3.7k)$	+1.8%/+1.1%
	(100, 150)	−0.15	−0.04	$(4.2k, 3.3k)$	+1.8%/+2.6%
	(100, 200)	−0.54	−0.48	$(4.2k, 2.7k)$	+1.8%/+6.1%
	(150, 100)	−0.26	−0.07	$(3.8k, 3.7k)$	+3.2%/+1.1%
	(150, 150)	−0.19	+0.01	$(3.8k, 3.3k)$	+3.2%/+2.6%
	(150, 200)	−0.45	−0.48	$(3.8k, 2.7k)$	+3.2%/+6.1%
	(200, 100)	−0.09	+0.08	$(3.1k, 3.7k)$	+6.9%/+1.1%
	(200, 150)	−0.09	+0.19	$(3.1k, 3.3k)$	+6.9%/+2.6%
	(200, 200)	=	+0.21	$(3.1k, 2.7k)$	+6.9%/+6.1%
$10k$	Baseline	34.02	79.46	$(8.8k, 6.6k)$	22.99/22.25
	(100, 100)	+0.15	+0.15	$(5.1k, 4.3k)$	+3.4%/+3.1%
	(100, 150)	−0.10	+0.10	$(5.1k, 3.0k)$	+3.4%/+9.5%
	(100, 200)	−0.17	+0.19	$(5.1k, 2.3k)$	+3.4%/+14.5%
	(150, 100)	−0.17	+0.11	$(3.6k, 4.3k)$	+10.2%/+3.1%
	(150, 150)	−0.20	+0.24	$(3.6k, 3.0k)$	+10.2%/+9.5%
	(150, 200)	−0.23	+0.10	$(3.6k, 2.3k)$	+10.2%/+14.5%
	(200, 100)	−0.12	+0.07	$(2.8k, 4.3k)$	+15.9%/+3.1%
	(200, 150)	−0.11	+0.14	$(2.8k, 3.0k)$	+15.9%/+9.5%
	(200, 200)	−0.17	+0.04	$(2.8k, 2.3k)$	+15.9%/+14.5%
$14k$	Baseline	33.94	79.47	$(12.0k, 8.9k)$	22.09/21.56
	(100, 100)	−0.39	−0.37	$(4.6k, 3.8k)$	+10.4%/+8.9%
	(100, 150)	−0.20	−0.14	$(4.6k, 2.6k)$	+10.4%/+16.0%
	(100, 200)	−0.30	−0.23	$(4.6k, 2.0k)$	+10.4%/+21.7%
	(150, 100)	−0.13	+0.03	$(3.1k, 3.8k)$	+18.7%/+8.9%
	(150, 150)	−0.44	−0.23	$(3.1k, 2.6k)$	+18.7%/+16.0%
	(150, 200)	−0.22	+0.03	$(3.1k, 2.0k)$	+18.7%/+21.7%
	(200, 100)	−0.21	+0.03	$(2.4k, 3.8k)$	+25.5%/+8.9%
	(200, 150)	−0.41	−0.20	$(2.4k, 2.6k)$	+25.5%/+16.0%
	(200, 200)	−0.26	+0.03	$(2.4k, 2.0k)$	+25.5%/+21.7%

Table 1: A subset of experimental results for the split- and joint-vocabulary settings. For each BPE baseline and its trimmed counterparts, we report BLEU (Papineni et al., 2002) and COMET (Rei et al., 2020) (relative to the baseline), the *effective vocabulary* size $(\hat{\mathbb{B}}_s, \hat{\mathbb{B}}_t)$, which is the size of the resulting vocabularies after trimming with the given thresholds, and *sequence length*, the average tokens-per-sentence in the tokenized test corpora (and the relative percent increase for the trimmed models). For both BLEU and COMET, the worst performing model in each setting is double underlined and the best performing model is underlined.

2 Experiments

To determine the effect of vocabulary trimming, we use the IWSLT14 German→English translation task (Cettolo et al., 2014). For all experiments, we use the same underlying `transformer-iwslt` architecture from `fairseq` (Ott et al., 2019), and only vary the embedding and decoding layers of the model by changing the tokenizer's source and target vocabulary sizes, $\mathbb{B}_s$ and $\mathbb{B}_t$ (or $\mathbb{B}_j$ for the *joint*-vocabulary setting), and source and target thresholds, $\mathbb{T}_s$ and $\mathbb{T}_t$, respectively. For the joint-vocabulary setting, a single tokenizer was formed by setting a vocabulary size and training the tokenizer on the concatenation of the source and target corpora. This baseline tokenizer was used to form separate source and target trimmed tokenizers.

As seen in Table 1, which contains a subset of our experimental results, while subword trimming reduces parameter count (by shrinking the embedding and decoding layers), it does not consistently improve performance and it causes an increase in average tokenized sequence length. In a sweep test, we found $(6k, 6k)$ to be the best performing split-vocabulary baseline and $7k$ and $10k$ to be the best performing joint-vocabulary baselines. For each of these configurations, trimming nearly always decreases BLEU, sometimes dramatically.

On the other hand, $(10k, 10k)$ was found to be the worst performing split-vocabulary baseline. Trimming this baseline increased BLEU, but not enough to match the better performing baseline models. For another baseline, $(8k, 8k)$, trimming did not consistently improve or degrade BLEU.

COMET shows a slight positive trend in most settings. In all but one case, trimming with a threshold of $(100, 100)$ lead to an improvement in over the baseline. Curiously, in the $10k$ joint-vocabulary setting, the trimmed models all have higher COMET scores than the baseline, while all but one have lower BLEU scores.

We conclude that vocabulary trimming should be done with caution, as it does not consistently improve performance, can heavily degrade performance, and comes at the cost of longer sequence lengths. This conclusion is based on the results from Table 1, as well as our much more expansive set of experimental results not listed here, which include many more ablation studies and a replication on the much larger Europarl English→French dataset (Koehn, 2005).

The complete results and code to reproduce them will be made public in our forthcoming full article.

References

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th IWSLT evaluation campaign. In *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 2–17, Lake Tahoe, California.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal archive*, 12:23–38.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

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, Philadelphia, Pennsylvania, USA. 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.

Rico Sennrich, Alexandra Birch, Anna Currey, Ulrich Germann, Barry Haddow, Kenneth Heafield, Antonio Valerio Miceli Barone, and Philip Williams. 2017. The University of Edinburgh's neural MT systems for WMT17. In *Proceedings of the Second Conference on Machine Translation*, pages 389–399, Copenhagen, Denmark. Association for Computational Linguistics.

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.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy. Association for Computational Linguistics.

On the Limits of Multi-modal Meta-Learning with Auxiliary Task Modulation Using Conditional Batch Normalization

Jordi Armengol-Estapé[*†1] **Vincent Michalski**[*2,3] **Ramnath Kumar**[4]
Pierre-Luc St-Charles[2] **Doina Precup**[2,5,7] **Samira Ebrahimi Kahou**[2,6,7]
[1]University of Edinburgh [2]Mila [3]Université de Montréal [4]Google Research
[5]McGill University [6]University of Calgary [7]CIFAR
jordi.armengol.estape@ed.ac.uk

Abstract

Few-shot learning aims to learn representations that can tackle novel tasks given a small number of examples. Recent studies show that cross-modal learning can improve representations for few-shot classification. More specifically, language is a rich modality that can be used to guide visual learning. In this work, we experiment with a multi-modal architecture for few-shot learning that consists of three components: a classifier, an auxiliary network, and a bridge network. While the classifier performs the main classification task, the auxiliary network learns to predict language representations from the same input, and the bridge network transforms high-level features of the auxiliary network into modulation parameters for layers of the few-shot classifier using conditional batch normalization. The bridge should encourage a form of lightweight semantic alignment between language and vision which could be useful for the classifier. However, after evaluating the proposed approach on two popular few-shot classification benchmarks we find that a) the improvements do not reproduce across benchmarks, and b) when they do, the improvements are due to the additional compute and parameters introduced by the bridge network. We contribute insights and recommendations for future work in multi-modal meta-learning, especially when using language representations.

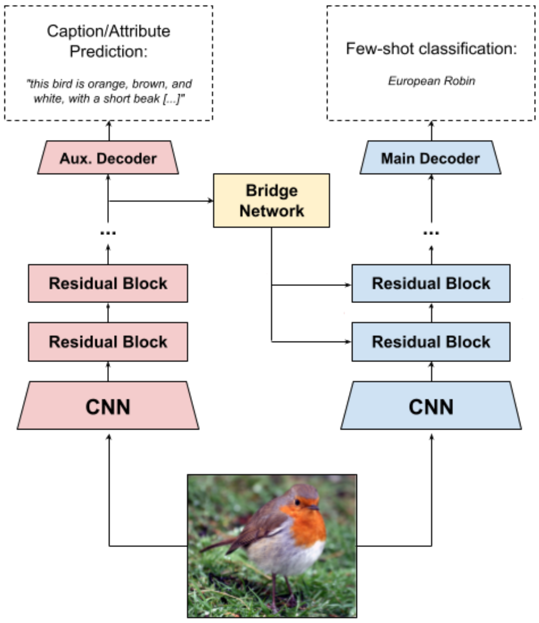


Figure 1: Architectural overview of the method we experimented with. It consists of three components: a classifier, an auxiliary network, and a bridge network. The few-shot classifier and auxiliary network receive the same input example. The bridge network transforms high-level features of the auxiliary network into modulation parameters for layers of the few-shot classifier through conditional batch normalization.

1 Introduction

It is widely recognized that humans can learn new concepts based on very little supervision, i.e. with few examples (or "shots"), and generalize these concepts to unseen data (Lake et al., 2011). Recent advances in deep learning on the other hand have mostly relied on datasets with large amounts of labeled examples, primarily due to overfitting concerns in low data regimes. Although the development of better data augmentation and regularization techniques can alleviate these concerns, many researchers now assume that future breakthroughs in low data regimes will emerge from either transferring generic models pretrained on very large datasets with unsupervised objectives (Devlin et al., 2019; Brown et al., 2020), or from meta-learning, i.e. "learning-to-learn". Here, we study the problem of learning-to-learn in few shots by using an embedding space in which we perform classification using a similarity metric. In this meta-

[*]Equal contribution.
[†]Work done while interning at Mila.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 51–59
June 20, 2024 ©2024 Association for Computational Linguistics

learning setting, a model is trained on a handful of labeled examples at a time under the assumption that it will learn how to correctly project examples of different classes and generalize this knowledge to unseen labels at test time.

Although this setting is often used to illustrate the remaining gap between human capabilities and machine learning, we could argue that the lack of context poses a serious disadvantage to machine learning models. Indeed, these models typically work based on a single-pass analysis while humans can first look at and understand contextual information before trying to interpret new classes (Swingley, 2010). It has been observed many times in the past that training models with contextual information such as auxiliary modalities can help build a more robust task-independent feature space (Ruder, 2017; Elliott et al., 2016; Radford et al., 2021). Auxiliary tasks however often require large support datasets with good label distributions and a delicate adjustment of network capacity to really help improve performance on the main task (Alonso and Plank, 2016). Multi-modal information can be difficult to process using a simple backbone architecture due to the varied structure and high-level nature of some typically used modalities, although recent Transformer-based works have shown it's possible, albeit costly (Jaegle et al., 2021). We refer to the Appendix A for a more comprehensive study of the related work.

We propose studying whether multitask learning with multi-modal objectives could be beneficial for few-shot learning even with commonly-used low-capacity feature extraction backbones, and without weight sharing between the main and auxiliary tasks. We study a way to condition multiple layers of our main feature extractor using an embedding produced by an entirely separate auxiliary network working on the same input data. The conditioning is applied to normalization layer parameters using a bridge network and it helps specialize the representations produced by the main feature extractor without affecting its architecture. Our idea here is to mimic the way humans can leverage context to help solve the recognition problem by combining low-level and high-level cues. In other words, we allow the main feature extractor to decide ahead of time what it should focus on based on task-level contextual knowledge. The proposed model architecture is illustrated in Figure 1. In contrast with previous works that also studied feature extraction conditioning and multi-modal learning, our approach is simple and can be applied to any feature extractor with batch normalization layers. The bridged-parallel-network design we propose also simplifies the feature alignment process since both branches process the same input data. Finally, the need for only a single input modality at test time leads to a more practical design for downstream applications.

However, after evaluating the proposed approach on two popular few-shot classification benchmarks we find that a) the improvements do not reproduce across benchmarks, and b) when they do, the improvements are due to the additional compute and parameters introduced by the bridge network. We contribute insights and recommendations for future work in multi-modal meta-learning, especially when using language representations.

2 Proposed method

In this section, we formulate conditional batch normalization in the context of few-shot learning. We propose a model, SimpAux, with two feature extractors that predict high-level (language-based) attributes of images as well as their semantic class. The embeddings of the attribute prediction pipeline (or "auxiliary" pipeline) are used to condition the batch normalization layers of the main visual feature extractor, which is based on a ProtoNet architecture (Snell et al., 2017). More specifically, we use ProtoNet++ improvement introduced in Oreshkin et al. (2018), with a Resnet-12 (He et al., 2016), which is a common choice in few-shot learning settings (e.g. Oreshkin et al., 2018; Jiang et al., 2019). The conditioning happens through a bridge connection composed of dense layers that translates the auxiliary embedding into batch normalization statistics. These three components are shown in Figure 1 and are described in the following subsections. Note that we use the same input modality (imagery) for the auxiliary and main feature extractors. However, our method is not limited to this modality: it was primarily chosen for compatibility with existing datasets. We refer the reader to Appendix B for a review of the fundamental ideas required to better understand our proposed few-shot learning solution from a technical standpoint.

2.1 Auxiliary visual processing

The auxiliary network in our proposed approach is agnostic of the main network's architecture and task. To simplify comparisons with a wider number of few-shot learning methods and to improve practicality, we formulate this network as a second visual processing pipeline that converts the same images fed to the main network into different embeddings. The multi-modal nature of our overall design comes from the supervised task used to learn the auxiliary network's embeddings: its goal is to predict language-based information from the images. More specifically, we experimented with predicting a) Attributes, available in datasets such as CUB-200-2011 (Wah et al., 2011), with cross-entropy, soft F1, or multi-label soft margin loss functions, and b) caption embeddings, with cosine similarity loss on the sentence embeddings emitted by SentenceBERT (Reimers and Gurevych, 2019). We ended up using multi-label soft margin loss as it was simpler and the other approaches did not provide significant improvements. However, for the datasets for which attributes were not available, we resorted to the sentence embeddings approach. As for the auxiliary model architecture itself, we also use a ResNet-12 as we do for the classifier.

2.2 Conditioning bridge

The role of the conditioning bridge is to transform the embeddings generated by the auxiliary network into an array of γ and β parameters that can be used in the various batch normalization layers of the main network. In contrast with late representation fusion strategies, e.g. the one of De Vries et al. (2017), this strategy allows for the early modulation of the main feature extraction pipeline with the high-level semantic information extracted from the auxiliary pipeline. Our hypothesis is that this information provides adequate context to dynamically adapt the main feature extractor while keeping its original architecture intact (and thus simple).

Since the distribution of the input representation varies at each layer of that network, the normalization parameters also need to be unique for each layer. We define our bridge as a multilayer perceptron (MLP) with a fixed intermediate representation size and an output size that corresponds to twice the total size of batch normalization layers in the main network (to account for both γ and β).

3 Experimental results

We evaluate SimpAux against the baseline, ProtoNet++, (the improved version of ProtoNets suggested in Oreshkin et al. (2018)) on two popular few-shot learning benchmarks, CUB-200-2011 (Wah et al., 2011) and mini-ImageNet (Vinyals et al., 2016) in 5-shot learning settings, using attributes for CUB and embeddings on synthetic captions for Mini-Imagenet for the auxiliary visual processing network. We refer to Appendix C for additional implementation details.

Table 1 shows the results of ProtoNet++ and SimpAux on CUB 5-shot. Our model clearly outperforms the baseline by a margin of around 1.5 points in accuracy.

Model	**Accuracy (%)**
ProtoNet++	88.5 ± 0.5
SimpAux	90.0 ± 0.7

Table 1: Accuracy on CUB. Each model was trained with five random seeds. Reported is the mean accuracy with 95% confidence intervals on 600 randomly generated test episodes.

These positive results on CUB showed the promise of the proposed approach. However, in the case of Mini-Imagenet 5-shot, in Table 2 we can see the results of ProtoNet++ and SimpAux on Mini-Imagenet. In this case, the baseline slightly outperforms the proposed method, but recall that here we are using synthetic captions.

Model	**Accuracy (%)**
ProtoNet++	75.4 ± 0.4
SimpAux	74.9 ± 0.1

Table 2: Accuracy on mini-ImageNet. Each model was trained with five random seeds. Reported is the mean accuracy with 95% confidence intervals on 600 randomly generated test episodes.

Finally, to test the hypothesis that the reason why our approach outperforms the baseline in CUB but not in ImageNet is the quality of the captions, we design an ablation study. We introduce a variation of SimpAux in which we use the exact same bridge network, but without input from the auxiliary network, to see whether the improvements are actually coming from the captions information or the additional compute and parameters from the bridge network. We find that there is no significant improvement over this variant when using the captions, suggesting that the improvement comes from the additional compute and the parameters provided by the bridge network.

4 Discussion and recommendations

From the experimental results, we conclude that a) the improvements provided by SimpAux do not reproduce across benchmarks, and b) when these improvements do indeed take place, they seem to be due to the additional compute and parameters provided by the bridge network.We hypothesize three non-mutually exclusive reasons why image captioning as auxiliary task modulation via conditional batch normalization did not consistently improve the results: 1) a lack of quality of the image captions, attributes, or caption embeddings, 2) the limited impact of the conditional batch normalization approach, and 3) the difficulty of learning the auxiliary task. While improving the quality of captions, attributes and caption embeddings with better annotations or more powerful models could alleviate 1), the following recommendations and observations look at other aspects involved in this work.

Caution when evaluating systems with auxiliary multi-modal information. Training models with contextual information such as auxiliary modalities have been shown to build a more robust task-independent feature space (Ruder, 2017; Elliott et al., 2016; Radford et al., 2021). However, spurious improvements with multi-modal data are not new. For instance, Elliott (2018) empirically raises doubts about whether existing multi-modal translation systems, combining visual and textual data, actually make use of the visual information. Similarly, we have seen the other way around: it is perfectly possible to outperform a unimodal baseline with a multi-modal one without actually making use of the textual information; SimpAux's improvements in CUB were due to the additional parameters introduced by the bridge network. Thus, we recommend extra care when concluding that multi-modal information helps in a certain task, which is definitely possible but could be due to other factors.

Importance of implementation details. We experimented with different activation functions, including ReLU (Agarap, 2018), SELU (Klambauer et al., 2017), and SiLU (Hendrycks and Gimpel, 2016; Ramachandran et al., 2017). We found that SiLU consistently yielded slightly better results across benchmarks and settings. Ensuring that weight decay was not applied to bias parameters, which is not the default behavior in PyTorch (Paszke et al., 2019), also turned out to be key to reproducing few-shot works originally implemented in Tensorflow (Abadi et al., 2015).

Hyperparameter search. In the hyperparameter search, we generally observed consistent results. However, we also observed a few outliers, which can be particularly extreme under certain settings in few-shot learning, and if used as empirical evidence, could totally change the conclusions. Thus, we reiterate the need for reporting averages and variances instead of the results of a single run, and also recommend caution at extracting certain conclusions when performing large-scale hyperparameter searches, as noted by Picard (2021).

Advantages of the proposed architecture. Our network architecture decouples task-specific branches: its bridge acts as a gate that selects relevant hints from the auxiliary network to influence the classification network. It is simpler than previous works that also studied feature extraction conditioning and multi-modal learning, and by design it requires a single input modality at test time, which simplifies practical deployments. SimpAux's architectural considerations are orthogonal to other few-shot learning research lines, and could be combined with them. Thus, we believe that, despite the limited success in the meta-learning setting, these architectural advantages could be a source of inspiration for future work.

Language-informed representations and few-shot learning. Without episodic learning, Radford et al. (2021) showed that language-informed visual representations can be successfully learned with large-scale supervised contrastive pretraining. Their approach, CLIP, obtains high-performance at zero-shot classification. Leveraging their pretrained encoders could be interesting in the context of bootstrapping episodic learning with auxiliary tasks. It would however be difficult to guarantee that the classes used in few-shot settings have not been observed by CLIP during pretraining.

5 Conclusion

In this work, we have studied a new multi-modal architecture for few-shot learning consisting of an image classifier, an auxiliary network trained with image captions, and a modulating network based on conditional batch normalization to connect the two. While initially promising, we have observed the limits of this approach and how these limits could inform future research.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Abien Fred Agarap. 2018. Deep learning using rectified linear units (relu). *CoRR*, abs/1803.08375.

Héctor Martínez Alonso and Barbara Plank. 2016. When is multitask learning effective? semantic sequence prediction under varying data conditions. *arXiv preprint arXiv:1612.02251*.

Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. 2018. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*.

Luca Bertinetto, João F Henriques, Jack Valmadre, Philip Torr, and Andrea Vedaldi. 2016. Learning feed-forward one-shot learners. In *Advances in neural information processing systems*, pages 523–531.

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.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. 2019. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*.

Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. 2022. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*.

Harm De Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. 2017. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems*, pages 6594–6604.

Tristan Deleu, Tobias Würfl, Mandana Samiei, Joseph Paul Cohen, and Yoshua Bengio. 2019. Torchmeta: A Meta-Learning library for PyTorch. Available at: https://github.com/tristandeleu/pytorch-meta.

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.

Guneet S Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. 2019. A baseline for few-shot image classification. *arXiv preprint arXiv:1909.02729*.

Desmond Elliott. 2018. Adversarial evaluation of multimodal machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2974–2978, Brussels, Belgium. Association for Computational Linguistics.

Desmond Elliott, Douwe Kiela, and Angeliki Lazaridou. 2016. Multimodal learning and reasoning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*.

Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. 2017. Exploring the structure of a real-time, arbitrary neural artistic stylization network. *arXiv preprint arXiv:1705.06830*.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*, volume 1. MIT press Cambridge.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415.

Xun Huang and Serge Belongie. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. 2021. Perceiver IO: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*.

Xiang Jiang, Mohammad Havaei, Farshid Varno, Gabriel Chartrand, Nicolas Chapados, and Stan Matwin. 2019. Learning to learn with conditional class dependencies. In *International Conference on Learning Representations*.

Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. 2017. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Brenden Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua Tenenbaum. 2011. One shot learning of simple visual concepts. In *Proceedings of the annual meeting of the cognitive science society*, volume 33.

Tsendsuren Munkhdalai, Xingdi Yuan, Soroush Mehri, and Adam Trischler. 2018. Rapid adaptation with conditionally shifted neurons. In *International Conference on Machine Learning*, pages 3664–3673. PMLR.

Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. 2018. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pages 721–731.

Frederik Pahde, Oleksiy Ostapenko, Patrick Jä Hnichen, Tassilo Klein, and Moin Nabi. 2019. Self-paced adversarial training for multimodal few-shot learning. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 218–226. IEEE.

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.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2017. Film: Visual reasoning with a general conditioning layer. *arXiv preprint arXiv:1709.07871*.

David Picard. 2021. Torch.manual_seed(3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision. *CoRR*, abs/2109.08203.

Limeng Qiao, Yemin Shi, Jia Li, Yaowei Wang, Tiejun Huang, and Yonghong Tian. 2019. Transductive episodic-wise adaptive metric for few-shot learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3603–3612.

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*.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. 2017. Searching for activation functions. *CoRR*, abs/1710.05941.

Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning.

Scott Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. 2016. Learning deep representations of fine-grained visual descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 49–58.

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*. Association for Computational Linguistics.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. 2018. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems*, pages 2483–2493.

Eli Schwartz, Leonid Karlinsky, Rogerio Feris, Raja Giryes, and Alex M Bronstein. 2019. Baby steps towards few-shot learning with multiple semantics. *arXiv preprint arXiv:1906.01905*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087.

Daniel Swingley. 2010. Fast mapping and slow mapping in children's word learning. *Language learning and Development*, 6(3):179–183.

Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. 2020. Rethinking few-shot image classification: a good embedding is all you need? *arXiv preprint arXiv:2003.11539*.

Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. 2020. Cross-domain few-shot classification via learned feature-wise transformation. *arXiv preprint arXiv:2001.08735*.

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.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. *arXiv preprint arXiv:1606.04080*.

Risto Vuorio, Shao-Hua Sun, Hexiang Hu, and Joseph J Lim. 2019. Multimodal model-agnostic meta-learning via task-aware modulation. In *Advances in Neural Information Processing Systems*, pages 1–12.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. 2011. The caltech-ucsd birds-200-2011 dataset.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):1–34.

Chen Xing, Negar Rostamzadeh, Boris Oreshkin, and Pedro OO Pinheiro. 2019. Adaptive cross-modal few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4847–4857.

Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. 2020. Few-shot learning via embedding adaptation with set-to-set functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8808–8817.

Fang Zhao, Jian Zhao, Shuicheng Yan, and Jiashi Feng. 2018. Dynamic conditional networks for few-shot learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–35.

Imtiaz Masud Ziko, Jose Dolz, Eric Granger, and Ismail Ben Ayed. 2020. Laplacian regularized few-shot learning. *arXiv preprint arXiv:2006.15486*.

A Related Work

Network conditioning. Normalization layers have been used many times in the past as a means to influence the behavior of deep feature extractors. For example, early works in arbitrary style transfer studied how modulating instance normalization parameters could align representations across styles that are not already known at run time (Huang and Belongie, 2017; Ghiasi et al., 2017). The flexibility gained by this modulation strategy has been adopted to tackle many other problems where feature extractors must dynamically change their behavior at run-time. For example, De Vries et al. (2017) and Perez et al. (2017) use conditional normalization layers to manipulate feature extractors in a selective manner for visual question answering and reasoning tasks. In the few-shot learning literature, Oreshkin et al. (2018) apply a form of normalization conditioning for task-dynamic feature extraction. In their case, instances are first encoded with an "unconditioned" feature extractor, and the resulting embeddings are used to condition the same feature extractor in a subsequent pass. In contrast, we base our conditioning on auxiliary labels and formulate a single-pass inference process. We also do not impose any constraints on the architecture of the main or auxiliary networks, meaning one can be much smaller than the other if required by the limited size of the dataset.

Note that there are also alternative conditioning strategies for few-shot learning paradigms that do not involve normalization layers. For example, embeddings can be directly modulated by a second network stage that analyzes the contextual information from the task (Ye et al., 2020; Qiao et al., 2019). Popular feature extractor architectures can also be slightly modified by adding conditionally shifted neurons to adapt representations using context at prediction time (Munkhdalai et al., 2018). Alternatively, the entire parameter set of various convolutional layers inside the feature extractor can be inferred at prediction time using a parallel network (Bertinetto et al., 2016, 2018; Zhao et al., 2018). A recent approach has also been proposed by Chen et al. (2022) to adapt large-scale multi-modal transformer-based backbones. The downside to these solutions is the dependency on large networks that must learn complex modulation operations from the task context, or the use of a memory bank on which an attention mechanism can operate. In contrast, normalization conditioning is a more lightweight approach that is easier to learn in small data regimes due to the reduced complexity of the modulation factors (i.e. the normalization statistics).

Recent trends in few-shot learning. There have been far too many strategies proposed to tackle few-shot learning for us to inventory them here. For a survey and a modern taxonomy, we refer the reader to the work of Wang et al. (2020). Instead, we note that many researchers over the years have highlighted the lack of a universal evaluation methodology for these methods. Recent independent efforts have shown that many "state-of-the-art" solutions are actually quite fragile and can be outperformed by simple baselines when evaluated and compared properly (Chen et al., 2019; Dhillon et al., 2019;

Tian et al., 2020). All of these works found that simple CNN backbones trained using a cross entropy loss and then optionally fine-tuned on test time queries can deliver competitive performance with respect to recent models. Transductive learning using test time queries in particular has been recently re-explored as an effective solution for few-shot learning (Dhillon et al., 2019; Ziko et al., 2020). Such findings highlight that more research effort should be spent on model-agnostic robustness improvements and less on the introduction or tuning of new model architectures as well as their training regimes. Our work falls in line with this idea while also promoting the use of multi-modal labels for improved few-shot learning.

As for multi-modal few-shot learning itself: it is not a new approach to the problem, but it is also not a popular one, as typical benchmarks only focus on using only imagery as input. Nonetheless, multiple strategies have been proposed to help deal with data scarcity in few-shot learning. For example, Pahde et al. (2019) feed image captions to a generative model during training to obtain additional images of the target classes. Their method however relies on several pre-trained and notably hard-to-train model components. Xing et al. (2019) and Schwartz et al. (2019) also leverage caption data but instead combine visual and semantic representations to improve class discrimination in metric space. In contrast to our work, they rely on parallel feature extraction pipelines that are combined in a "late fusion" fashion, whereas we propose a way to modulate the entirety of any visual pipeline architecture with semantic information. Vuorio et al. (2019) applies a similar modulation idea to the model-agnostic, meta-learning (MAML) framework of Finn et al. (2017). In their case, they rely on the modulation layers proposed by Perez et al. (2017) to condition their main task network. Tseng et al. (2020) follow the same strategy to deal with domain generalization issues in few-shot learning. In comparison, our proposed auxiliary network is trained in a supervised cross-modal setting where its embeddings are used to modulate our main network. Also, since we apply modulation through batch normalization, our approach can handle data samples that do not possess auxiliary labels or captions.

B Background

Here, we review some of the fundamental ideas required to understand our proposed few-shot learning solution.

B.1 Episodic few-shot learning and ProtoNets

In episodic few-shot learning, an "episode" is represented as an N-way, K-shot classification problem where N is the number of examples per class and K the number of unique class labels. During training, the data in each episode is provided as a support set $S = \{(\boldsymbol{x}_{1,1}, \boldsymbol{y}_1), ..., (\boldsymbol{x}_{N,K}, \boldsymbol{y}_N)\}$ where $\boldsymbol{x}_{i,j} \in \mathbb{R}^D$ is the i-th instance of the j-th class, and $\boldsymbol{y}_j \in \{0,1\}^K$ is its corresponding one-hot labeling vector. The goal in each episode is to optimize a function f that classifies new instances provided through a "query" set Q which contains instances of the same classes as S. This task is difficult because N is typically very small (e.g. 1 to 10), the classes change every episode, and the actual test set used to evaluate a model does not contain classes that were seen in support sets during training.

We build our solution on top of Prototypical Networks (ProtoNets; Snell et al., 2017), as it is now accepted as a good yet simple baseline. According to (Chen et al., 2019), it is more robust than other recent few-shot learning approaches and it generalizes well across various dataset domains. ProtoNets tackle few-shot learning by learning an embedding space where each class is represented by a cluster, or *prototype*. A prototype $\boldsymbol{c_k} \in \mathbb{R}^M$ for a class k is simply defined as the mean of the instance embeddings that belong to k, that is:

$$\boldsymbol{c_k} = \frac{1}{S_k} \sum_{(\boldsymbol{x}_{i,j}, \boldsymbol{y}_i) \in S_k} f(\boldsymbol{x}_{i,j}), \tag{1}$$

where S_k is the support subset of all instances that belong to class k, and f is a learned function. Next, the probability of assigning a new instance x to a class k is computed via the softmax of the distance to all class prototypes:

$$p(y = k|\boldsymbol{x}) = \frac{\exp -d(f(\boldsymbol{x}), \boldsymbol{c}_k)}{\sum_{k'} \exp -d(f(\boldsymbol{x}), \boldsymbol{c}_{k'})}, \tag{2}$$

for any given distance function $d : \mathbb{R}^M \times \mathbb{R}^M \mapsto [0, +\infty)$.

B.2 Batch normalization and conditioning

Batch normalization was proposed by (Ioffe and Szegedy, 2015) as a solution to speed up training by reducing the problem of coordinating weight updates across the different layers of a model. In short, batch normalization performs a reparameterization on the intermediate representations of a model so that assumptions regarding their spread and distribution in subsequent layers will be less affected by stochastic updates. More specifically, given a batch of n feature maps $B = \{\boldsymbol{z}_1, ..., \boldsymbol{z}_n\}$ with C channels each, batch normalization performs channel-wise reparameterization using

$$\mathrm{BN}(\boldsymbol{z}_{l,c}|B, \boldsymbol{\gamma}, \boldsymbol{\beta}) = \gamma_c \cdot \frac{\boldsymbol{z}_{l,c} - \mu_c}{\sigma_c^2 + \epsilon} + \beta_c, \quad (3)$$

where $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are vectors of learned channel-wise parameters, ϵ is a constant used for numerical stability, and μ_c and σ_c^2 are the mean and variation values computed across batch and spatial dimensions of B.

Many researchers now recognize that batch normalization has beneficial side-effects on the landscape of the optimization problem (Goodfellow et al., 2016; Santurkar et al., 2018). These benefits have lead to the rapid adoption of this technique across the majority of new and popular model architectures. Consequently, the important role and ubiquitous nature of batch normalization make it an interesting target for the conditioning of models using auxiliary data. This idea was first introduced by De Vries et al. (2017): they inject visual concepts from natural language in a visual processing pipeline for VQA by manipulating batch normalization parameters. These parameters are influenced by the embeddings produced with a recurrent network. One advantage of this approach is that it can help learn how to dynamically specialize a model at test time without drastically increasing its overall number of learnable parameters. This advantage is very interesting in the context of few-shot learning where only small datasets prone to overfitting are considered.

C Other implementation details

Our ProtoNet backbone is the improved version of the original method (coined ProtoNet++) suggested by Oreshkin et al. (2018) that includes residual connections between convolution layers (Resnet-12). We implement the models and data loaders with PyTorch (Paszke et al., 2019) and Torchmeta (Deleu et al., 2019), a meta-learning library. We experimented with different activation functions, and SiLU (Ramachandran et al., 2017) yielded the best results.

In our experiments, we use the CUB-200-2011 (Wah et al., 2011) and mini-ImageNet (Vinyals et al., 2016) datasets. For CUB, we use the split of Chen et al. (2019) and also experiment with the captions collected by Reed et al. (2016). For mini-ImageNet, we use the setting proposed by Ravi and Larochelle (2016), with synthetic captions generated using an open-source implementation of a Transformer (Vaswani et al., 2017) for image captioning.[1]

Our implementation is publicly available on Github.[2]

[1] https://github.com/saahiluppal/catr
[2] https://github.com/jordiae/simpaux-release

Pointer-Generator Networks for Low-Resource Machine Translation: Don't Copy That!

Niyati Bafna, Philipp Koehn, and David Yarowsky
Johns Hopkins University, Center for Language and Speech Processing
{nbafna1,phi,yarowsky}@jhu.edu

Abstract

While Transformer-based neural machine translation (NMT) is very effective in high-resource settings, many languages lack the necessary large parallel corpora to benefit from it. In the context of low-resource (LR) MT between two closely-related languages, a natural intuition is to seek benefits from structural "shortcuts", such as copying subwords from the source to the target, given that such language pairs often share a considerable number of identical words, cognates, and borrowings. We test Pointer-Generator Networks for this purpose for six language pairs over a variety of resource ranges, and find weak improvements for most settings ($<$ 1 BLEU). However, analysis shows that PGNs do not show greater improvements for closely-related vs. more distant language pairs, or for lower resource ranges, and that the models do not exhibit the expected usage of the mechanism for shared subwords. Our discussion of the reasons for this behaviour highlights several general challenges for LR NMT, such as modern tokenization strategies, noisy real-world conditions, and linguistic complexities. We call for better scrutiny of linguistically motivated improvements to NMT given the black-box nature of Transformer models, as well as for a focus on the above problems in the field.

1 Introduction and Motivation

While state-of-the-art (SOTA) Transformer models (Vaswani et al., 2017) for NMT work well for high-resource language pairs, their performance degrades in low-resource situations (Koehn and Knowles, 2017; Sennrich and Zhang, 2019; Kim et al., 2020; Haddow et al., 2022); this means that most languages in the world cannot benefit from mainstream advances and models (Joshi et al., 2020). There is therefore a clear appeal to developing simple architectural mechanisms for these models that are targeted at yielding improvements in data-scarce scenarios, while interfering minimally with mainstream preprocessing, tokenization, and training pipelines.

कारन, खरास, उतरले, मिलल
कारण, खराश, उतरे, मिले
reason, soreness, descended-pl, met

Figure 1: Translation equivalents for Bhojpuri (top) and Hindi (bottom), demonstrating subword overlap.

In the context of a low-resource language (LRL), we are often interested in translation to and from a closely related HRL, which possibly has linguistic genealogical, regional, and cultural ties with the LRL,[1] in order to make the abundant content in HRLs available in related LRLs. We expect that closely related languages share considerable overlap at the subword level from cognates, borrowings and shared vocabulary (see examples in Figure 1). Given the absence of large parallel corpora for our language pair, we aim to leverage this shared knowledge across source and target, intuitively, to provide "easier" routes for our MT model from source to target sentence.

Pointer Generator Networks (PGNs; See et al. (2017)) are a mechanism which allow the model, for every output token produced, to either copy some token from the input ("point") or "generate" a token as per usual from the vocabulary. PGNs have been used for a variety of problems, described in Section 2, often targeted at repeated spans of text in the input and output; however, as we far as we know, this is the first work to study its applicability to LR NMT. In this case, we hypothesise that the pointing mechanism will show advantages for rare shared subwords, for which the best strategy may be to copy them to the output.

We introduce a PGN mechanism into a Transformer-based NMT architecture, and test

[1] This is the case, for example, for several languages of the Arabic continuum, all closely related to relatively high-resource Modern Standard Arabic, and languages of the Turkic and Indic language continua.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 60–72
June 20, 2024 ©2024 Association for Computational Linguistics

its performance for 6 language pairs over 4 low-resource training ranges. We work with Hindi-Bhojpuri (hi-bh), Spanish-Catalan (es-ca), and French-Occitan (fr-oc), representing closely-related pairs, Hindi-Marathi (hi-mr), a relatively more distant pair,[2] and Spanish-English (es-en) and French-German (fr-de), representing further distant pairs. We expect that PGN will help most for (1) lower-resource scenarios (2) more closely-related language pairs (3) sentence pairs with higher subword overlap. While PGN shows improvements in certain settings, our comparative analysis of the benefits of PGNs across the three dimensions above shows clear lack of evidence for the these hypotheses. Further, our visualizations of the PGN mechanism also indicate that observed benefits do not come from intended sources. We discuss various factors that contribute to this failure, highlighting fundamental challenges for LR NMT, such as noisy datasets, mainstream tokenization practices best suited for high-resource scenarios, as well as linguistic and orthographic complexities that may obfuscate underlying source-target similarities.[3]

2 Related Work

Pointer Networks were introduced to solve problems that involved permuting the input, such as the Traveling Salesman Problem and the complex hull problem (Vinyals et al., 2015). Their use in NLP has been largely been for monolingual summarization, where the target may naturally contain identical spans from the source. Cheng and Lapata (2016) present a complex hierarchical LSTM-based model for summarization, which directly extracts sentences from the text and words from sentences. Gulcehre et al. (2016) use pointer networks in RNN-based sequence-to-sequence models for summarization and machine translation, training their model explicitly to use the pointing mechanism for uncommon words. Gu et al. (2016) and See et al. (2017) also incorporate variants of pointer-generator networks into RNN-based sequence-to-sequence learning for summarization. Prabhu and Kann (2020) applied PGNs to the task of grapheme-to-phoneme conversion via an explicit source-target mapping. Zhang et al. (2021) proposed a pointer-disambiguator-copier (PDC) system for dictionary-enhanced high-resource NMT, using source word translations as potential candidates for the copying mechanism, with a disambiguator component to select appropriate senses.

Our work is the first to examine the applicability of PGNs as facilitators in the low-resource MT scenario, looking to exploit linguistic relationships between the source and target in the absence of external resources. We work with Transformer-based NMT, and make no changes to standard BPE tokenization schemes or training objectives (unlike Gulcehre et al. (2016) and Zhang et al. (2021)). This is so that our findings are most relevant in today's paradigm of generalized strategies for end-to-end multilingual MT; our mechanism can be easily plugged into and trained with any modern (multilingual) MT pipeline.

3 Model

The PGN model provides two routes to the model for predicting any target token: copying from the source or generating from the vocabulary (Prabhu and Kann, 2020). Copy and generate distributions at step t are mixed using a learned parameter p^t_{copy}, to obtain the final probability distribution P^t for the target token.

$$p^t_{copy} = \sigma(\mathbf{W}^T(\mathbf{c}^t \oplus \mathbf{d}^t \oplus \mathbf{s}^t) + \mathbf{B})$$
$$\mathbf{P}^t = p^t_{copy} \cdot \mathbf{P}^t_{\mathbf{c}} + (1 - p^t_{copy}) \cdot \mathbf{P}^t_{\mathbf{g}}$$

Here, $\mathbf{c}^t$ is the context vector, calculated as $\mathbf{c}^t = (\mathbf{a}^t)^T\mathbf{e}^t$, where $\mathbf{a}^t$ represents cross-attention vector, and $\mathbf{e}^t$ contains the encoder hidden states. $\mathbf{d}^t$ and $\mathbf{s}^t$ contain the decoder's final hidden states and input respectively, $\oplus$ denotes concatenation, and $\mathbf{W}$ and $\mathbf{B}$ are learned weights and a bias vector respectively. $\mathbf{P}^t_{\mathbf{c}}$ and $\mathbf{P}^t_{\mathbf{g}}$ represent the copy and generate distributions (softmaxed logits) respectively at step t. We use cross-attention weights over source tokens for the copy logits, and standard decoder outputs for generate logits.

4 Experiments

Datasets and languages We used the WikiMatrix (`wm`) corpus (Schwenk et al., 2019) for `es-en`, `es-ca`, `fr-de` and `fr-oc`. For `es-ca`, we also report results on synthetic Europarl (`ep`) parallel data

[2] Hindi, Bhojpuri, and Marathi belong to the Indic branch of the Indo-European family. Hindi and Bhojpuri further belong to the Shaurasenic sub-branch and are closer lexically and grammatically to each other and other languages on or close to the Hindi Belt such as Punjabi, Rajasthani, Haryanvi, and Maithili, than Hindi is to Marathic languages and dialects; this is supported by lexical and other studies of cross-lingual similarity (Sengupta and Saha, 2015; Mundotiya et al., 2021; Bafna et al., 2022). See Glottolog (`https://glottolog.org/resource/languoid/id/cont1248`) for the phylogenetic tree.

[3] `https://github.com/niyatibafna/pgns-for-lrmt`

(Koehn, 2005).[4] For hi-mr, we used the CVIT-PIB corpus (Philip et al., 2021), and for the low-resource pair hi-bh, we use the NLLB corpus (Schwenk et al., 2021; Team et al., 2022; Heffernan et al., 2022). See Table 1 for dataset heuristics. Note the higher per token overlap as expected for our closely-related group as compared to the others. The hi-bh sentences are extremely short, and share fewer tokens than expected: in this case, this reflects badly parallel data.[5]

Experimental Settings For all language pairs, we performed experiments on dataset subsets of $5k$, $15k$, $30k$, and $60k$ sentences and test sets of 5000 sentences, trained until convergence, with tokenizer size 16000. We computed baseline results (NMT) on standard encoder-decoder NMT. All PGN and NMT models use 6 encoder and decoder layers, 4 attention heads, and a hidden size of 512.

5 Results and Discussion

Improvement patterns See results in Table 2.[6] Our results are not directly comparable to those in the literature due to differences mentioned in Section 2 and the size of training bitext ($2M$ in (Gulcehre et al., 2016), $1M$ in Zhang et al. (2021) vs. our maximum resource setting of $0.06M$).[7] We see weak improvements in a majority of settings; however, counter to intuition, PGN does not show a clear advantage for closely-related as compared to more distant pairs, or for lower-resource settings.

Controlled test sets We test the motivating hypothesis that PGN models is likely to benefit sentence pairs with higher subword overlap. We rank sentence pairs in our test set by percentage of shared subwords in source and target, and construct test subsets with low and high shared-subword density from the top and bottom 500 sentences respectively. However, in Table 3, we see that in fact that PGN performs slightly worse than NMT on both extremes, indicating that observed benefits over the entire test set do not come from subword overlap.

Usage of the copy mechanism We record values of p_{copy} to track the model's usage of the copy mechanism. While p_{copy} values are relatively high[8] for copied subwords, numerals, and proper nouns, we often see that they they are also high for seemingly random subwords.[9] We also do not see a relationship between the p_{copy} value of a target token and the entropy of the cross-attention distribution for that token.

A reasonable intuition about PGN training generalization is that in the absence of any information, the model will default to copying, since this is likely to do better on average than guesses over the entire vocabulary, and that eventually, it will learn to generate language-specific subwords, memorising the relevant strategy for given subwords in encoder states (used to calculate p_{copy} as shown in Section 3). However, our visualizations of cross-attention and p_{copy} usage throughout training show no evidence of this generalization strategy. It's possible that since initial cross-attention distributions are noisy, and most subwords are not direct copies, the model is discouraged from copying early on; it's also possible that the model finds it easier or trivial to encode copied source-target equivalents via the "generate" mechanism and does not need an explicit copier, given that it must additionally learn which subwords should be copied. We discuss potential reasons for this below. In general, it appears that the model uses the copy mechanism to encode a task that is not easily interpretable, possibly resulting in the observed small improvements over some datasets.

Tokenization In theory, the copier would learn best if the tokenizer behaved in a morphologically principled manner.[10] However, BPE tokenization generally results in subword splits that may not reflect shared stems in word equivalents (Ataman and Federico, 2018).[11]

A natural idea here may be an investigation of morphologically inspired tokenizers (Pan et al., 2020; Ortega et al., 2020; Chen and Fazio, 2021).

[4]The Europarl dataset was automatically translated into Catalan; taken from `https://github.com/Softcatala/Europarl-catalan`.

[5]See Appendix A for more details on datasets.

[6]We report spBLEU since our approach attempts to benefit performance on shared subwords.

[7]For a rough idea: Zhang et al. (2021) report gains in MT of $1.5 - 2.5$ BLEU.

[8]Note that it is difficult to comment on absolute values of p_{copy}. The copying distribution is normalized over the sentence length whereas generate distributions are normalized over the vocabulary; even low values of p_{copy} will considerably affect the mixed distribution.

[9]See Appendix C for visualizations of this behaviour. Examples with counter-intuitively high values of p_{copy}: quiero-vull (es-ca), behad-atishay (hi-mr).

[10]e.g. given khaya-khalla (ate) in Hindi and Marathi, we ideally want kha ##ya and kha ##lla. This will allow the common stem kha to be copied over, while the language specific inflection subwords can be generated.

[11]e.g. our trained tokenizer contains both propuesto (es) and proposat (ca) instead of sharing the subword prop.

	hi-mr	hi-bh	es-en	es-ca (ep)	es-ca (wm)	fr-de	fr-oc
Avg. common tokens per sent pair	2.51	1.29	2.81	6.77	7.17	1.76	5.06
Avg. common tokens per target token	0.12	0.16	0.13	0.26	0.29	0.10	0.22
Avg. source sentence length	28.54	6.34	23.43	26.86	25.38	19.10	24.09
Avg. target sentence length	20.82	7.98	20.77	26.26	24.64	16.93	23.43

Table 1: Statistics on commons source-target tokens in our datasets. wm: WikiMatrix, ep: Europarl.

		5K	15K	30K	60K	Avg. Δ
hi-mr	NMT	3.4	7.4	11.9	16.3	
	PGN	3.4	**7.9**	**12.6**	**18.4**	+0.8
<u>hi-bh</u>	NMT	**5.6**	6.1	9.7	**15.8**	
	PGN	4.0	**8.3**	**11.4**	12.7	-0.2
es-en	NMT	**9.8**	**30.4**	38.3	41.7	
	PGN	9.4	30.0	**38.5**	**42.2**	0.0
<u>es-ca(wm)</u>	NMT	**35.4**	50.9	**54.3**	56.4	
	PGN	34.7	**51.2**	54.1	**57.1**	0.0
<u>es-ca(ep)</u>	NMT	**62.6**	70.6	73.2	73.6	
	PGN	62.5	**71.6**	**74.0**	**74.2**	+0.6
fr-de	NMT	3.5	10.8	19.5	27.2	
	PGN	**3.7**	**11.1**	**20.1**	27.2	+0.3
<u>fr-oc</u>	NMT	24.7	42.2	45.5	**48.7**	
	PGN	**24.8**	**43.4**	**46.4**	48.5	+0.5

Table 2: spBLEU across dataset sizes (#sents). Closely-related pairs are <u>underlined</u>. wm: WikiMatrix, ep: Europarl.

		hi-mr	<u>hi-bh</u>	es-en	<u>es-ca</u>	fr-de	<u>fr-oc</u>	Avg. Δ
L	NMT	7.6	10.4	**26.6**	**44.9**	**11.9**	**35.5**	
	PGN	**8.2**	**13.5**	24.4	44.6	10.3	35.3	-0.7
H	NMT	28.9	**22.1**	69.8	**72.4**	**52.2**	**65.2**	
	PGN	**29.3**	18.3	**69.9**	71.6	51.6	64.3	-1.0

Table 3: spBLEU scores on test sets with low (L) and high (H) density of shared source-target subwords.

However, we generally see inconclusive, at best marginal, benefits of such tokenizers over BPE in modern neural MT (Macháček et al., 2018; Domingo et al., 2019; Mielke et al., 2021), especially those relying on unsupervised morphological segmentation, e.g., with Morfessor (Creutz and Lagus, 2007) in the absence of morphological analysers. These ideas have not been incorporated into mainstream tokenization strategies.

Recent work attempts to solve this general problem by looking at maximisation of shared subwords in multilingual tokenizers (Chung et al., 2020; Zheng et al., 2021; Liang et al., 2023); it's possible that such strategies will dovetail well with PGN mechanisms if widely adopted in the future.

Linguistic complexities While closely-related language show high (subword) vocabulary overlap, word equivalences may be obscured by sound change and orthographic systems; if these changes are word-internal, then even an ideal tokenizer will see different stems/tokens in the source and target.[12] Further, we may see that a word that has a cognate in its sister language is translated to a non-cognate, due to semantic drift, or differences in idiom or usage norms in the two languages, e.g. kitaab-pustak (hi-mr), resulting in non-identical subword equivalences. These phenomena are often unpredictable and unsystematic; even if not, they are not trivial to model into tokenization or architectural strategies for MT.

See Appendix B for experiments with minor variants of our approach dealing with pretrained encoder/decoder initialization, tokenizer size, choice of attention head, and identical source-target settings.

6 Challenges for LR NMT

Incorporating knowledge of linguistic relationships among closely related data-imbalanced language pairs offers a natural strategy for mitigating data scarcity in mainstream NMT between regional languages, and it is crucial to understand the challenges in this realm. We show that while the PGN mechanism offers an intuitive theoretical shortcut for translation between closely related languages, its performance in practice is limited, potentially by the combined effect of noisy real-world datasets containing non-literal translations, the behaviour of

[12] e.g. vi**sh**was-**bis**was (hi-bh, sound change), websi**te**-Webs**eite** (en-de, orthographic system)

standard tokenizers, as well as linguistic complexities beyond simplified ideas of shared vocabulary and cognates. These are inevitable hurdles to any project that attempts to use structural linguistic knowledge to benefit NMT performance.

Further, we show that despite showing benefits in certain settings over the entire test set, the PGN mechanism does not perform as expected on target phenomena. The generalization mechanisms of blackbox Transformer-models are not well understood and may not be easily guided by linguistic intuition: we underline the importance of verifying that improvements are coming from the intended places rather than good starts or extra parameters.

Finally, our analysis hints that PGN-like shortcuts may not be worth offering in the first place: "easy" equivalences, a natural target of linguistic interventions, may not be the bottleneck for LR NMT. Instead, it is more likely that the true bottlenecks are handling precisely the above challenges, i.e. non-systematic differences, one-off phenomena, and real-world noise in low-resource conditions.

7 Limitations

While we show that our particular flavour of NMT incorporating PGNs does not provide fundamental benefits for low-resource NMT, this is naturally not to say that an improved variant of this idea would not work better. There are several potential ways forward arising from our discussion of the reasons for the failure of our method in Section 5: for example, using morphological segmentation for tokenization to increase subword overlap, or using priors for p_{copy} so that it is encouraged for shared subwords. Previous work provides different kinds of help to the copier: for example, (Gulcehre et al., 2016) explicitly train the copier to copy unknown words with a separate training objective. However, as we mention in Section 1, our motivation lay in designing a simple architectural mechanism that can be easily integrated into mainstream (multilingual) NMT pipelines to make them more capable for low-resource MT, without requiring much additional language-pair specific attention to training paradigms or tools such as morphological analysers and bilingual lexicons, which are in any case of poor quality for low-resource languages. We restrict our negative result to the scope set up by this motivation.

Further, our results are limited to the 6 language pairs that we experimented on. While we simulate identical low-resource conditions for all our language pairs, we clearly see the difference in absolute performances on `hi-mr` or `hi-bh` as compared to the high-resource language pairs: the data for the latter are simply of much better quality. This demonstrates the need to experiment and present further results on non-simulated truly low-resource conditions, such as the hi-bh language pair studied here. Finally, this discussion is only relevant to translation between closely-related languages that share a script (although this is the predominant case), allowing for lexical similarity to be reflected by shared subwords.

8 Conclusion

In this work, we investigate the applicability of Pointer-Generator Networks in NMT, hypothesizing that an explicit copy mechanism will provide benefits for low-resource translation between closely related languages. We show that while we do observe weak improvements, these are not higher for closer-related languages, sentence pairs with higher overlap, or lower resource ranges, contrary to intuition. Our discussion of potential reasons for the failure of this approach highlights several general challenges for low-resource NMT, such as mainstream tokenization strategies, noisy data, and non-systematic linguistic differences.

References

Duygu Ataman and Marcello Federico. 2018. An evaluation of two vocabulary reduction methods for neural machine translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 97–110.

Niyati Bafna, Josef van Genabith, Cristina España-Bonet, and Zdeněk Žabokrtský. 2022. Combining noisy semantic signals with orthographic cues: Cognate induction for the Indic dialect continuum. In *Proceedings of the 26th Conference on Computational Natural Language Learning (CoNLL)*, pages 110–131, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*.

William Chen and Brett Fazio. 2021. Morphologically-guided segmentation for translation of agglutinative low-resource languages. In *Proceedings of the 4th Workshop on Technologies for MT of Low Resource Languages (LoResMT2021)*, pages 20–31, Virtual. Association for Machine Translation in the Americas.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany. Association for Computational Linguistics.

Hyung Won Chung, Dan Garrette, Kiat Chuan Tan, and Jason Riesa. 2020. Improving multilingual models with language-clustered vocabularies. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4536–4546, Online. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(1):1–34.

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*.

Miguel Domingo, Mercedes Garcıa-Martınez, Alexandre Helle, Francisco Casacuberta, and Manuel Herranz. 2019. How much does tokenization affect neural machine translation?

Mikel L Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M Tyers. 2011. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. *arXiv preprint arXiv:1603.08148*.

Barry Haddow, Rachel Bawden, Antonio Valerio Miceli Barone, Jindřich Helcl, and Alexandra Birch. 2022. Survey of low-resource machine translation. *Computational Linguistics*, 48(3):673–732.

Kevin Heffernan, Onur Çelebi, and Holger Schwenk. 2022. Bitext mining using distilled sentence representations for low-resource languages.

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.

Raviraj Joshi. 2022. L3cube-hindbert and devbert: Pre-trained bert transformer models for devanagari based hindi and marathi languages. *arXiv preprint arXiv:2211.11418*.

Tanmai Khanna, Jonathan N Washington, Francis M Tyers, Sevilay Bayatlı, Daniel G Swanson, Tommi A Pirinen, Irene Tang, and Hèctor Alòs i Font. 2021. Recent advances in apertium, a free/open-source rule-based machine translation platform for low-resource languages. *Machine Translation*, 35(4):475–502.

Yunsu Kim, Miguel Graça, and Hermann Ney. 2020. When and why is unsupervised neural machine translation useless? In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 35–44, Lisboa, Portugal. European Association for Machine Translation.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

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.

Davis Liang, Hila Gonen, Yuning Mao, Rui Hou, Naman Goyal, Marjan Ghazvininejad, Luke Zettlemoyer, and Madian Khabsa. 2023. Xlm-v: Overcoming the vocabulary bottleneck in multilingual masked language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, page 13142–13152, Singapore. Association for Computational Linguistics.

Dominik Macháček, Jonáš Vidra, and Ondřej Bojar. 2018. Morphological and language-agnostic word segmentation for nmt.

Sabrina J. Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y. Lee, Benoît Sagot, and Samson Tan. 2021. Between words and characters: A brief history of open-vocabulary modeling and tokenization in nlp.

Rajesh Kumar Mundotiya, Manish Kumar Singh, Rahul Kapur, Swasti Mishra, and Anil Kumar Singh. 2021. Linguistic resources for bhojpuri, magahi, and maithili: Statistics about them, their similarity estimates, and baselines for three applications. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 20(6).

John E Ortega, Richard Castro Mamani, and Kyunghyun Cho. 2020. Neural machine translation with a polysynthetic low resource language. *Machine Translation*, 34(4):325–346.

Yirong Pan, Xiao Li, Yating Yang, and Rui Dong. 2020. Morphological word segmentation on agglutinative languages for neural machine translation.

Jerin Philip, Shashank Siripragada, Vinay P. Namboodiri, and C. V. Jawahar. 2021. Revisiting low resource status of indian languages in machine translation. In *Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD)*, page 178–187, Bangalore India. ACM.

Nikhil Prabhu and Katharina Kann. 2020. Making a point: Pointer-generator transformers for disjoint vocabularies. 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: Student Research Workshop*, pages 85–92.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2019. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. *arXiv preprint arXiv:1907.05791*.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. 2021. CCMatrix: Mining billions of high-quality parallel sentences on the web. 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 6490–6500, Online. Association for Computational Linguistics.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Debapriya Sengupta and Goutam Saha. 2015. Study on similarity among indian languages using language verification framework. *Advances in Artificial Intelligence*, 2015:2–2.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy. Association for Computational Linguistics.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

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. *Advances in neural information processing systems*, 30.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. *Advances in neural information processing systems*, 28.

Tong Zhang, Long Zhang, Wei Ye, Bo Li, Jinan Sun, Xiaoyu Zhu, Wen Zhao, and Shikun Zhang. 2021. Point, disambiguate and copy: Incorporating bilingual dictionaries 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 3970–3979.

Bo Zheng, Li Dong, Shaohan Huang, Saksham Singhal, Wanxiang Che, Ting Liu, Xia Song, and Furu Wei. 2021. Allocating large vocabulary capacity for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3203–3215, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Notes on Datasets

The **Hindi-Marathi** WikiMatrix dataset (Schwenk et al., 2021) has only 11k sentences so we use CVIT-PIB (Philip et al., 2021) instead. The CVIT-PIB corpus is automatically aligned using an iterative process that depends on neural machine translation into a pivot language and filtering heuristics. Eyeballing the data, we observe a considerable number of non-parallel or even entirely unrelated sentences simply containing some words in common - in general, this corpus is much more likely to contain rough paraphrases as opposed to literal translations.

For **Hindi-Bhojpuri**, NLLB seems to be the only available parallel text data for now (Tiedemann, 2012); this corpus has also been automatically crawled (Team et al., 2022). The Hindi-Bhojpuri NLLB dataset contains extremely short sentences as shown in Table 1; similarly to above, we observe a high level of noise and non-parallel data. Such datasets naturally do not provide the most favourable training conditions for the PGN models, which rely on literal translations containing shared subwords to teach the copier; however, they are realistic real-word conditions for truly low-resource languages, as we discuss in Section 6.

The WikiMatrix dataset (Schwenk et al., 2019), which we use for **Spanish-English**, **Spanish-Catalan**, **French-German**, and **French-Occitan** is automatically aligned from Wikipedia content in these languages.

The **Spanish-Catalan** synthetic Europarl bitext is created by automatically translating the Europarl dataset (Koehn, 2005) into Catalan using Apertium (Forcada et al., 2011; Khanna et al., 2021). While this data probably contains some noise due to MT errors and translationese, it's the most likely of all our datasets to contain literal, linear translations, and we include it as a testbed for this purpose. It is a generally easier dataset - this is clearly visible from spBLEU scores that our models achieve on it in Table 2.

B Minor Variations

Pretrained encoder and decoder We tried using a pretrained encoder and decoder at initialization of our model and tested this for `hi-mr` and `es-ca`. For the former, the encoder and decoder were initialized with Hindi BERT (Joshi, 2022), and for the latter, we used Spanish BERT (Cañete et al., 2020). These pre-trained models are language-specific instances of BERT (Devlin et al., 2018). Note that this means that we also used the pretrained tokenizers of these models, of sizes 52000 and 31002 for Hindi and Spanish respectively, that are only trained on the high-resource source languages; this leads to very poor tokenization in the target language. In general, this set of models take longer to converge due to their size, and show only minor differences in performance. Another related idea is to finetune NLLB or another multilingual MT model with an incorporated PGN; we did not try this given the lack of encouraging results from these experiments.

Single attention head We also tried using only a single attention head to calculate p_{copy} for target tokens, with the motivation that it was maybe better to nudge a single head to encode information about whether target token need to be copied, and leaving other heads to generate, as opposed to asking all heads to do both (which is the case when we average over heads). However, these models give almost identical results as in Table 2.

Smaller tokenizer We hypothesize that using a smaller tokenizer size will force more splits per token, increasing the chance that common stems will be reflected in shared subwords. Accordingly, we tried a tokenizer size of 8000 for `hi-mr` and `es-ca` for the $15k$ and $60k$ settings; however, performance degrades slightly (about -1 spBLEU on average) for both NMT and PGN approaches without affecting the relative trend.

This is not altogether surprising: reducing tokenizer size only increases the degree of splitting in words of a certain (lower) frequency range, rather than affecting the number of splits for all words uniformly. More importantly, while these hyperparameters are important to tune, statistical frequency-based tokenizers behave inherently differently from morphologically-inspired tokenizers, as discussed in Section 5, and it is not easy or perhaps possible to achieve a good approximation of the latter by playing with the hyperparameters of the former.

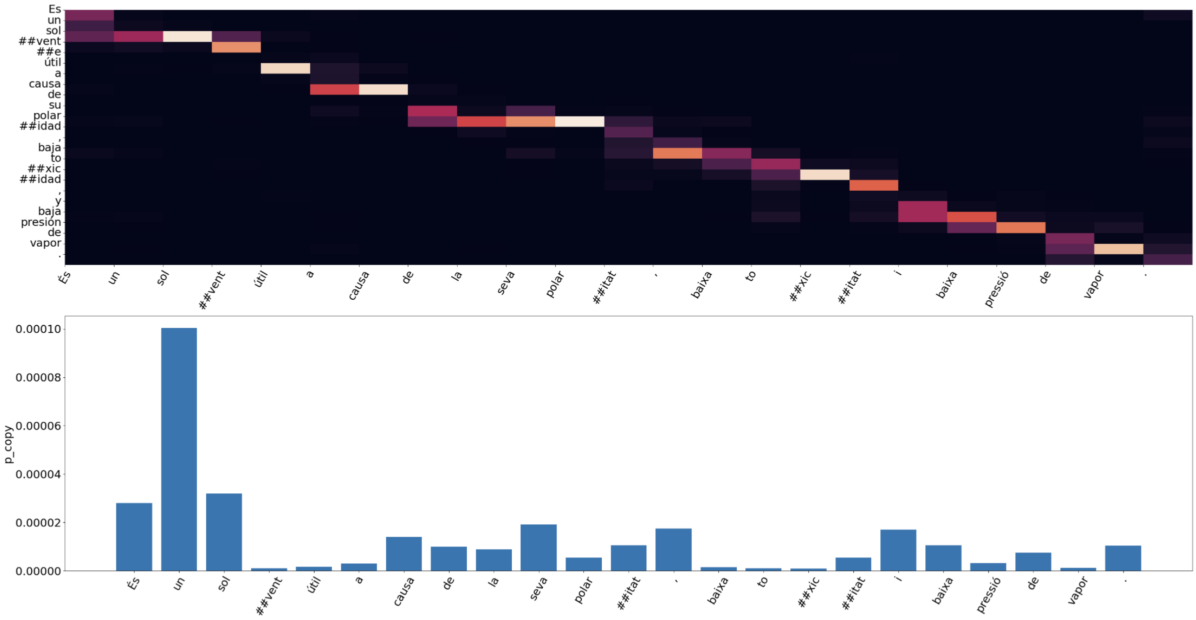
Identical source and target Finally, we also trained a Hindi-Hindi model, to remove the effects of noisy translations and non-ideal tokenization of source and target token sequences as discussed in Sections 4 and 5. In this setup, with 100% overlap, the models achieve high test scores (74 spBLEU) and converge to near-zero usage of the copy mechanism. Clearly, the model still prefers to encode the identity relationship using a generate mechanism.

Note that this setup is fundamentally different from our other scenarios - when all tokens are copied, the model no longer needs the distinction between two distinct processes (generating and copying), and therefore does not really need to learn how to make this decision. However, it is still illustrative in demonstrating that model generalization mechanisms, even for highly simplified or trivial tasks, are often not intuitive or human-interpretable.

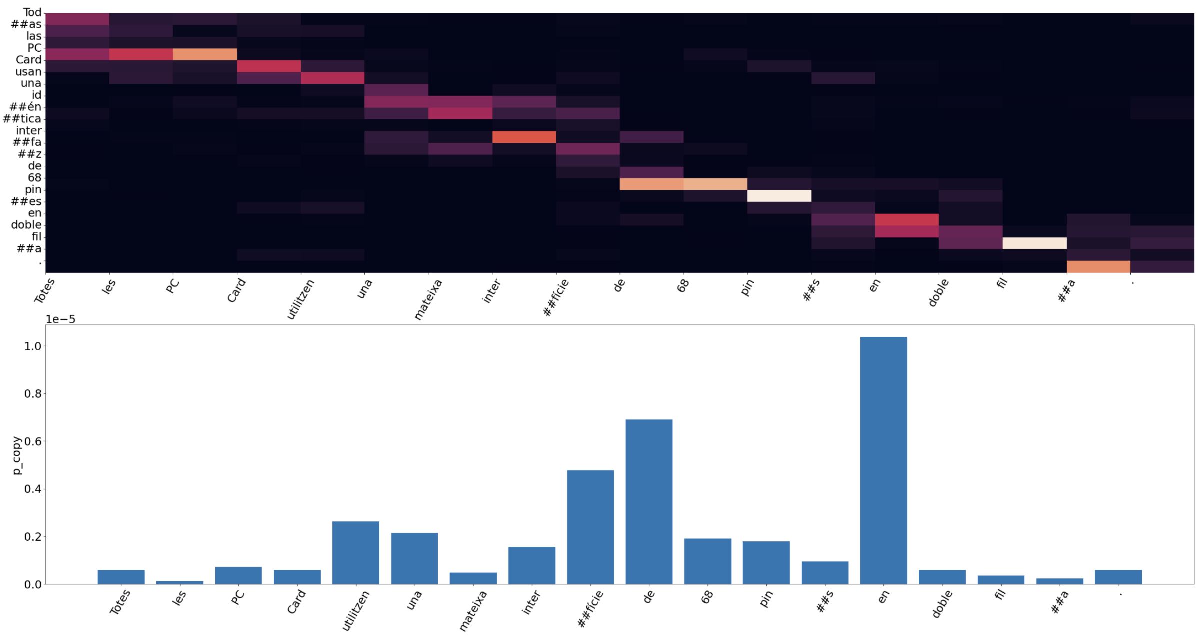
C Visualizations

See Figures 2, 3, 4, and 5 for visualizations of the PGN model's cross-attention distributions and values of p_{copy} per target token on randomly chosen source-target pairs using an early and late model training checkpoint. We observe that the model does use the copying mechanism as intended in many places, for common subwords (`udaar` in `hi-mr`, `un` in `es-ca`) as well as named entities (`Cour` in `fr-oc`), common borrowings (`computer` in `hi-bh`), numbers (`1970` in `fr-oc`) and punctuation. However, p_{copy} values are also relatively high sometimes for other seemingly random target tokens, e.g. `canton-costat` in `fr-oc`.

Note that the `hi-bh` source-target sentence pairs are not in fact translations of each other and exemplify the noise we discuss in Section 4 and Appendix A. The cross-attention distributions for the `es-ca` and `fr-oc` models are in general much better defined and able to attend to appropriate tokens (in these example visualizations as well as others that we looked at); this is a consequence of the better quality of the data and models in these languages.

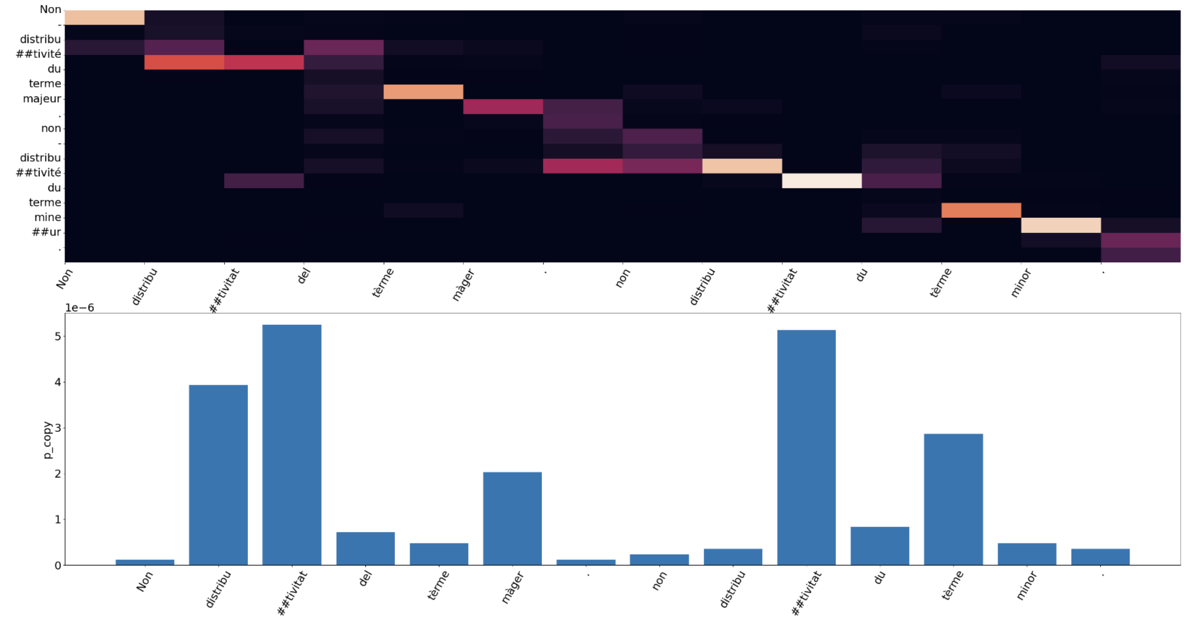


(a) Epoch 10

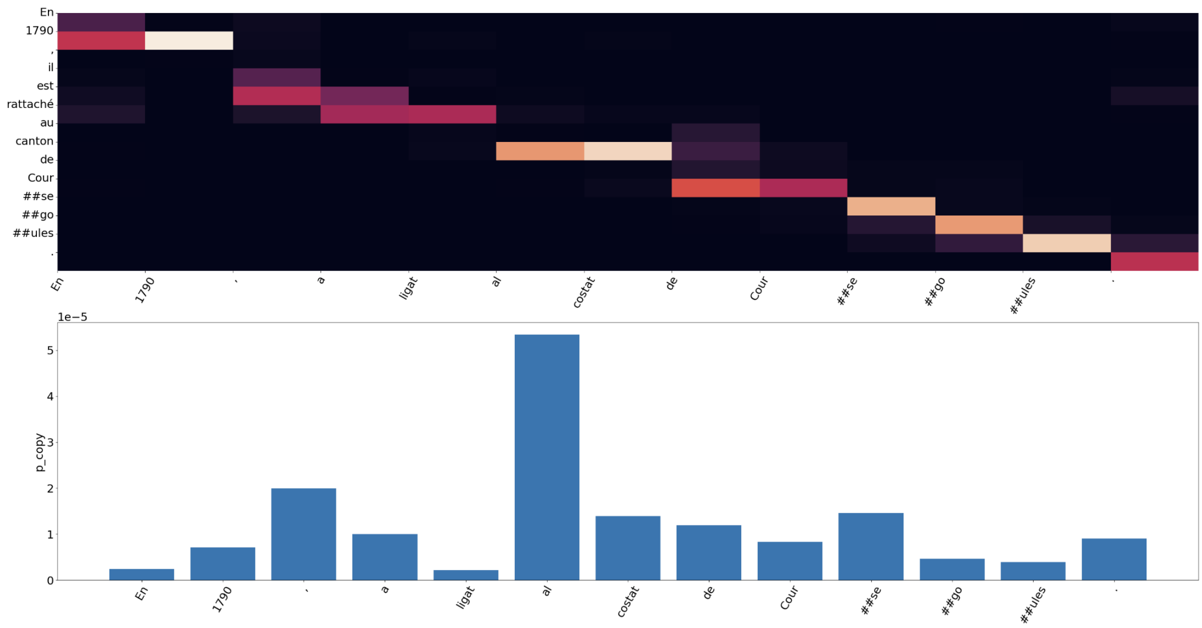


(b) Epoch 30

Figure 2: Model's cross-attention distributions and p_{copy} values for two sentence pairs for `es-ca(ep)`, $60k$ sentences

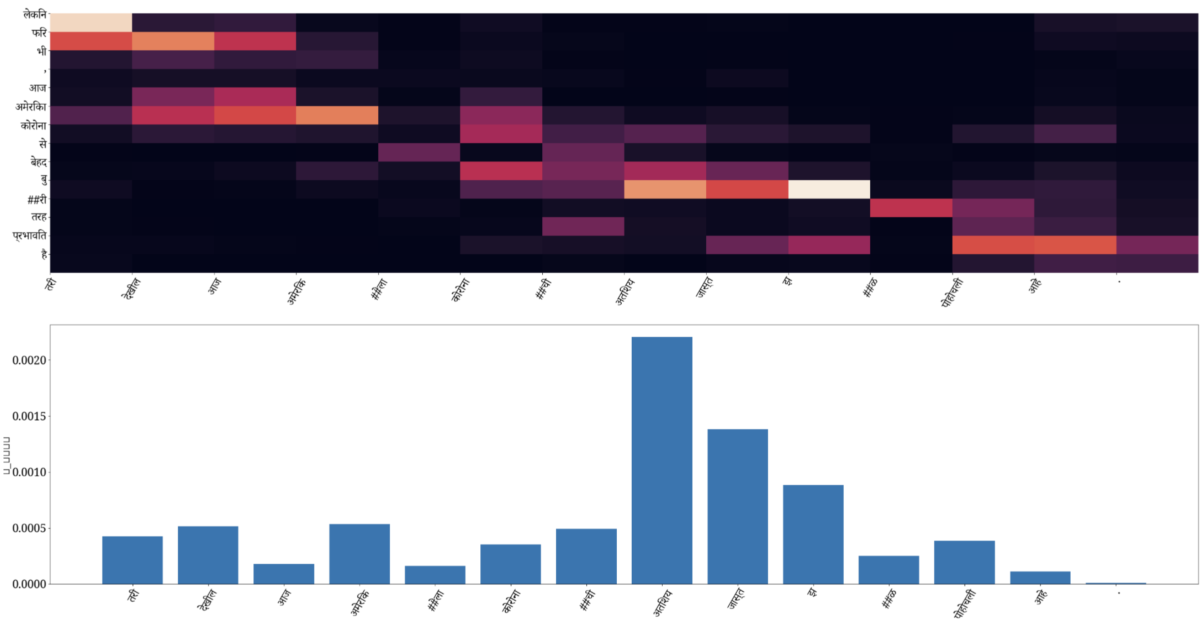


(a) Epoch 10

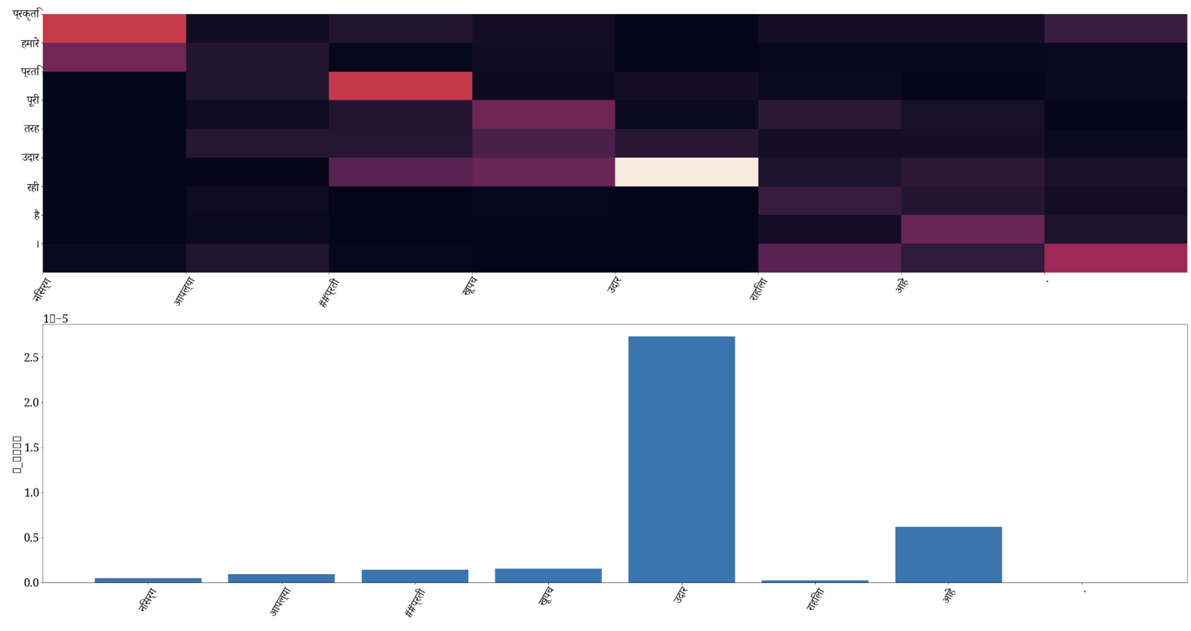


(b) Epoch 30

Figure 3: Model's cross-attention distributions and p_{copy} values for two sentence pairs for `fr-oc`, $60k$ sentences

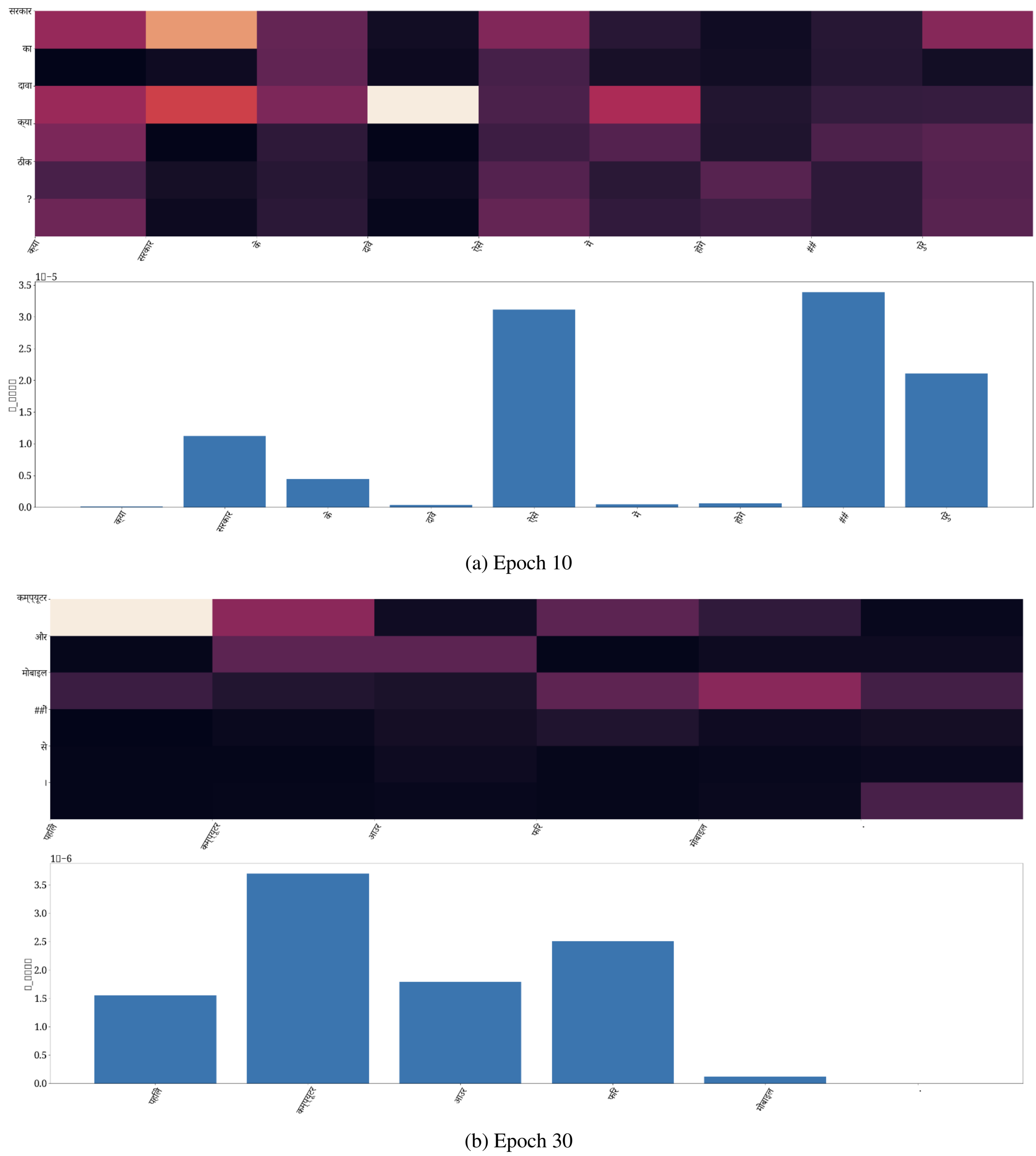


(a) Epoch 10



(b) Epoch 25

Figure 4: Model's cross-attention distributions and p_{copy} values for two sentence pairs for `hi-mr`, $60k$ sentences



(a) Epoch 10

(b) Epoch 30

Figure 5: Model's cross-attention distributions and p_{copy} values for two sentence pairs for `hi-bh`, $60k$ sentences

Imaginary Numbers! Evaluating Numerical Referring Expressions by Neural End-to-End Surface Realization Systems

Rossana Cunha, Osuji Cynthia Chinonso, João Gabriel Moura Campos,
Brian Timoney, Brian Davis, Fabio Cozman, Adriana Pagano and Thiago Castro Ferreira
Arts Faculty, Federal University Minas Gerais, Brazil
Adapt Research Centre, Dublin City University, Ireland
Escola Politécnica, University of São Paulo, Brazil
rossanacunha@ufmg.br chinonso.osuji@adaptcentre.ie joaogcampos@usp.br
brian.timoney3@mail.dcu.ie brian.davis@adaptcentre.ie fgcozman@usp.br
apagano@ufmg.br thiagocf05@ufmg.br

Abstract

Neural end-to-end surface realizers output more fluent texts than classical architectures. However, they tend to suffer from adequacy problems, in particular *hallucinations* in numerical referring expression generation. This poses a problem to language generation in sensitive domains, as is the case of robot journalism covering COVID-19 and Amazon deforestation. We propose an approach whereby numerical referring expressions are converted from digits to plain word form descriptions prior to being fed to state-of-the-art Large Language Models. We conduct automatic and human evaluations to report the best strategy to numerical superficial realization. Code and data are publicly available[1].

1 Introduction

The significant advances in deep learning for NLP and its enormous success in other text generation tasks, such as machine translation (Akhbardeh et al., 2021). As a result, approaches to surface realization of *data-to-text* systems have moved from traditional modular pipeline architectures (Reiter and Dale, 2000) to end-to-end ones. These systems transform a simple meaning representation into text without any explicit intermediate representations (Wen et al., 2015; Dušek and Jurčíček, 2016; Lebret et al., 2016; Gehrmann et al., 2018). While early neural data-to-text systems required a high amount of parallel training data, current state-of-the-art (SOTA) architectures, known as Large Language Models (LLMs) (Radford et al., 2019; Lewis et al., 2020; Raffel et al., 2020a), can deliver impressive results with less training, even excelling in zero-short or few-shot settings.

With respect to linguistic output, neural end-to-end surface realizers appear to generate more fluent text than classical pipeline architectures but are more likely to suffer from (semantic) adequacy problems, in particular, *hallucinations* (Ji et al., 2023), whereby the system produces text that contains information which is not present in the input representation. A particular hallucination problem that modern approaches seem to struggle with, unlike classical architectures, is numerical referring expression generation (Puduppully and Lapata, 2021; Wallace et al., 2019; Ji et al., 2023). For instance, let's hypothesize the case where a surface realizer produces the outcome: *"The country registered* ***458,098*** *cases of COVID-19"*, whereas the gold-standard reference points to *"The country registered* ***408,098*** *cases of COVID-19"*. Albeit there is only a single-digit difference between both texts (which can be overlooked by popular automatic quality metrics), the difference represents an arithmetic change of 50,000 and may lead readers to make drastic errors given the sensitivity of the context.

To the best of our knowledge, this problem has never been investigated in surface realization systems, despite having been addressed in other generation tasks such as text normalization (Zhang et al., 2019; Sproat, 2022), question-answering (Chen et al., 2021; Kim et al., 2022), and *text-to-speech* (Nikulásdóttir and Guðnason, 2019); tasks which also struggle to synthesize texts with numerical referring expressions represented by digits. One approach to circumvent the problem in *text-to-speech* systems is to normalise the input texts by converting numerical referring expressions from digits to plain word form descriptions prior to being fed into the system (Nikulásdóttir and Guðnason, 2019). Another technique used in Referring Expression Generation (REG) systems is slot-filling or *delexicalisation* where values like date, number, or constants are represented as a literal (Castro Ferreira et al., 2018; Cunha et al., 2020).

In the context of end-to-end surface realizers, this study raises two questions:

[1] https://github.com/BotsDoBem/LargeLM

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 73–81
June 20, 2024 ©2024 Association for Computational Linguistics

	B. Portuguese			English		
	Train	Dev	Test	Train	Dev	Test
Daily Deforestation	4,062	504	484	3,874	452	462
Month Deforestation	324	20	22	456	36	26
Daily Fire	942	108	108	–	–	–
COVID-19	1,064	122	108	–	–	–
Total	6,392	754	722	4,330	488	488

Table 1: Data Statistics.

INPUT
[DEFORESTATION_MONTH][INTENTS] TOTAL_DEFORESTATION (area="322.91"', location="deter-amz", month="4", year="2021") [HISTORY] [PARAGRAPH]

PORTUGUESE OUTPUT
O Instituto Nacional de Pesquisa Espaciais (INPE) informou que foram desmatados 322.91 km² na Amazônia Legal, em abril de 2021.

ENGLISH OUTPUT
The National Institute for Space Research (INPE) detected 322.91 sq km of deforestation in the Legal Amazon in April 2021.

Figure 1: Example of Portuguese and English Meaning Representation inputs and their corresponding outputs.

(RQ1) *How well do state-of-the-art end-to-end surface realizers generate numerical referring expressions?*
(RQ2) *Are numerical referring expressions better verbalized when represented by digits or text (spell-out form)?*

To answer these questions, we conducted automatic and human evaluations with three SOTA LLMs: GPT-2 (Radford et al., 2019), BART (Lewis et al., 2020), T5 (Raffel et al., 2020b), and their multilingual counterparts. These models were used to verbalize English and Brazilian Portuguese news about Amazon Deforestation, Fire Alerts, and COVID-19 cases using **four** different strategies, which we discuss in Section 3. Code and data will be publicly available.

2 Data

For training and evaluating the models, we used automatic-generated reports by *BotsDoBem*, a group of Twitter robot-journalists such as CoronaReporter[2] and DaMata[3], which publish news in Brazilian Portuguese and English. For Brazilian Portuguese, the dataset comprises of i) both daily and monthly reports on deforestation in the Legal Amazon area of Brazil (Rosa Teixeira et al., 2020), ii) daily reports about Fires in the Brazilian Biomes, as well as iii) COVID-19 cases in the country (Campos et al., 2020). For English, the dataset comprises of daily and monthly reports on deforestation in the Legal Amazon. Although automatically generated, these texts contain a high number of numerical referring expressions, making them suitable for our goal of evaluating how well neural end-to-end surface realizers generate numerical referring expressions. Table 1 introduces the number of instances per language and domain, split into training, development, and test sets. Each instance in the corpus consists of a meaning representation and a corresponding gold-standard verbalization in Brazilian Portuguese or English representing the sentence of a report. For both languages, the verbalizations were automatically generated by the pipeline system described in Rosa Teixeira et al. (2020) and Campos et al. (2020).

Figure 1 illustrates the structure of instances in both the English and Portuguese datasets, which consist of meaning representations starting with a tag representing the report domain, followed by a tag that marks the beginning of the sentence intents (e.g., `INTENTS`). Each intent in the meaning representation follows the *intent-attribute-value* schema. Finally, the tag `[HISTORY]` marks where the verbalization of the previous sentences in the paragraph of the target will be depicted. In the example, the tag `[PARAGRAPH]` means that the target sentence is at the beginning of the paragraph.

3 Numerical Referring Expressions

To evaluate the effectiveness of a neural end-to-end surface realizer in generating numerical expressions, we consider **two forms** of number representation: digits and word (*spell-out*) form descriptions. These are assessed in both the meaning representations and the verbalizations, resulting in a total of **four** distinct strategies:

1. Numbers represented by digits in the meaning representation and the reference texts (*no desc*);
2. Numbers are described in the input meaning representation in spell-out form and digits in the target references (*desc src*);
3. Numbers represented by digits in the meaning representations and spell-out form descriptions in the target references (*desc trg*); and
4. Numbers are described in a spell-out form in both the input meaning representations and target references (*desc*).

To exemplify, Table 2 depicts the **four** strategies of a pair of meaning representations and their corresponding English verbalizations. We utilized

[2] https://twitter.com/CoronaReporter
[3] https://twitter.com/DaMataReporter

	Numeric Referring Expressions			
Strategies	**Area**	**Month**	**Year**	**Input MR**
no desc	322.91	4	2021	In April 2021, 322.91 sq km of the Legal Amazon were deforested, according to data from the National Institute for Space Research (INPE)
desc src	three hundred and twenty-two point nine one	four	two thousand and twenty-one	In April 2021, 322.91 sq km of the Legal Amazon were deforested, according to data from the National Institute for Space Research (INPE).
desc trg	322.91	4	2021	In April *two thousand and twenty-one*, *three hundred and twenty-two point nine one* sq km of the Legal Amazon were deforested, according to data from the National Institute for Space Research (INPE).
desc	three hundred and twenty-two point nine one	four	two thousand and twenty-one	In April *two thousand and twenty-one*, *three hundred and twenty-two point nine one* sq km of the Legal Amazon were deforested, according to data from the National Institute for Space Research (INPE).

Table 2: The strategies and representations of the numeric referring expressions. Strategies are *highlighted*.

the Python library[4], *num2words*, to transform numerical digits into their textual counterparts. This library is effective for both English and Brazilian Portuguese languages.

4 Experiments

To address our first research question **(RQ1)**, we evaluate the performance of three LLMs in generating numerical references: I) GPT-2, ii) BART, and iii) T5 for English domains. Additionally, for Portuguese, we fine-tuned GPorTuguese-2 (Guillou, 2020), a Brazilian Portuguese version of GPT-2, as well as mBART-50 (Tang et al., 2020) and mT5 (Xue et al., 2021), which are the multilingual versions of BART and T5, respectively. These models were selected due to a more sustainable perspective of LLMs (Rillig et al., 2023) and the environmental implications of the new LLMs, such as ChatGPT (OpenAI, 2023) and BARD[5]. The model training process involved 30 epochs, a learning rate of 1e-5, a batch size of 1, 5 early stops, and a maximum token length of 300.

4.1 Automatic Evaluation

We computed the BLEU score (Papineni et al., 2002) of the system to analyze the generated texts' fluency automatically and whether errors in numerical referring expressions are reflected in its result.

4.2 Human Evaluation

To answer our research questions **(RQ1)** and **(RQ2)**, we performed a human evaluation against the outcomes of our evaluated approaches.

Method We perform the human evaluation following the methodology of Thomson and Reiter (2020), which aims to quantify the quality of automatically generated texts according to the following taxonomy of errors: *Incorrect Number*, *Incorrect Named Entity*, *Incorrect Word*, *Context*, *Not Checkable* and *Other*. Besides these categories, a *Fluency* error category was incorporated into the evaluation, which allowed raters to assess the output for issues related to text flow acceptability. We are primarily interested in the dimensions concerning the number errors i.e., *Incorrect Number* and *Incorrect Word*. We also drew on best practices concerning error analysis and reporting as described in van Miltenburg et al. (2021).

Data preparation and Annotation process Overall, we selected 20% of a stratified sample, comprising 852 instances of Brazilian Portuguese output (per strategy and model). Three linguistically proficient annotators assessed these instances. To ensure reliability, a duplicate batch was evaluated by the same three raters. For English, all 240 outputs (per strategy per model) were independently annotated by two linguistically proficient raters. This process followed a pilot annotation of 50 instances for each language to clarify any ambiguities in the annotation guidelines before the full annotation task. Brazilian and English annotators and/or raters are members of the research team.

It is worth noting that for the Portuguese dataset annotators evaluated different entries in the first and second batches, allowing for inter-rater agreement assessment. To reduce bias during double annotation, access to corresponding entries in different batches was not allowed. For both datasets, in line with Thomson and Reiter (2020) methodology, we removed any disagreement as a result of raters not following annotations guidelines.

5 Results

The error rates and BLEU scores for each numerical strategy and model for both English and Portuguese are presented in Table 3. Numerical errors were found to be the most common type across both languages. However, the numerical error rates

[4] https://pypi.org/project/num2words/
[5] https://bard.google.com/

Error Rate Full Results for English (EN) and Brazilian Portuguese (PT)

S	LM		Number		Named Entity		Word		Context		Uncheckable		Other		Fluency		BLEU	
	EN	PT	EN	PT	EN	PT	EN	PT	EN	PT	EN	PT	EN	PT	EN	PT	EN	PT
No Desc	T5	mT5	0.48	0.45	0.05	0.02	0.08	0.08	0.03	0.03	0.03	0.03	0.08	0.07	0.05	0.03	0.69	0.58
	GPT2	GPT2-pt	0.65	0.24	0.28	0.04	0.03	0.03	0.53	0.03	0.08	0.01	0.60	0.14	0.95	0.09	0.14	0.60
	BART	mBART	0.50	0.34	0.00	0.04	0.00†	0.00†	0.08	0.09	0.00†	0.01	0.00	0.07	0.15	0.10	0.61	0.51
	Avg.	**Avg.**	**0.54***	**0.34**	**0.11**	**0.04**	**0.03**	**0.04***	**0.21**	**0.05***	**0.03**	**0.02**	**0.23**	**0.09**	**0.38**	**0.07**	**0.48**	**0.56**
D. Source	T5	mT5	0.45†	0.37	0.00†	0.01	0.00†	0.08	0.00†	0.05	0.05	0.01	0.00†	0.04	0.08	0.02	0.69	0.59
	GPT2	GPT2-pt	1.00	0.19†	0.05	0.01	0.00†	0.02	0.03	0.12	0.00†	0.01	0.00†	0.08	0.23	0.03	0.41	0.61
	BART	mBART	0.48	0.28	0.00†	0.03	0.00†	0.01	0.05	0.01	0.00†	0.01	0.03	0.13	0.23	0.05	0.62	0.59
	Avg.	**Avg.**	**0.64**	**0.28***	**0.02***	**0.01**	**0.00***	**0.04**	**0.03***	**0.06**	**0.02**	**0.01**	**0.01**	**0.09**	**0.18**	**0.03***	**0.57**	**0.60**
D. Target	T5	mT5	0.95	0.87	0.00†	0.00†	0.00†	0.04	0.03	0.00†	0.00†	0.00†	0.00†	0.01†	0.00†	0.02†	0.87†	0.65
	GPT2	GPT2-pt	0.95	0.90	0.03	0.01	0.00†	0.04	0.13	0.11	0.13	0.00†	0.00†	0.08	0.18	0.05	0.35	0.64
	BART	mBART	0.90	0.79	0.05	0.06	0.08	0.09	0.05	0.09	0.03	0.00†	0.00†	0.10	0.18	0.09	0.60	0.61
	Avg.	**Avg.**	**0.93**	**0.85**	**0.03**	**0.02**	**0.03**	**0.06**	**0.07**	**0.07**	**0.05**	**0.00***	**0.00***	**0.06**	**0.12**	**0.05**	**0.60***	**0.64**
Desc	T5	mT5	0.93	0.90	0.00†	0.00†	0.03	0.05	0.03	0.02	0.00†	0.00†	0.00†	0.01	0.00†	0.06	0.66	0.68†
	GPT2	GPT2-pt	0.90	0.80	0.13	0.01	0.03	0.12	0.23	0.14	0.03	0.00†	0.05	0.03	0.25	0.15	0.28	0.67
	BART	mBART	1.00	0.89	0.00†	0.00	0.03	0.07	0.03	0.03	0.00†	0.00†	0.00†	0.03	0.05	0.15	0.58	0.65
	Avg.	**Avg.**	**0.94**	**0.87**	**0.04**	**0.00***	**0.03**	**0.08**	**0.09**	**0.06**	**0.01***	**0.00***	**0.02**	**0.02**	**0.10***	**0.12**	**0.50**	**0.67***

Table 3: Error rates and BLEU score for the 4 numerical strategies and 3 language models – Higher error rates denote more errors. Higher BLEU scores denote greater Fluency. ∗(Lowest error rate among strategies averages); †(Lowest error rate among model and strategy combinations); S (Strategies); and D (Desc).

Incorrect Number Error Rate

Strategies	LM	English (EN)			B. Portuguese (PT)		
		DM	DD	Overall	DM	DD	Overall
No Desc	T5/mT5	0.50	0.45†	0.48	0.55	0.18	0.36
	GPT2/GPT2-pt	0.65	0.65	0.65	0.00†	0.00†	0.00†
	BART/mBART	0.50	0.50	0.50	0.18	0.09	0.14
	Avg.	**0.55***	**0.53***	**0.54***	**0.24***	**0.09**	**0.17***
Desc Source	T5/mT5	0.45†	0.45†	0.45†	0.73	0.07	0.40
	GPT2/GPT2-pt	1.00	1.00	1.00	0.27	0.00†	0.14
	BART/mBART	0.50	0.45†	0.48	0.45	0.00†	0.23
	Avg.	**0.65**	**0.63**	**0.64**	**0.48**	**0.02***	**0.25**
Desc Target	T5/mT5	0.95	0.95	0.95	1.00	0.68	0.84
	GPT2/GPT2-pt	0.95	0.95	0.95	0.82	0.74	0.78
	BART/mBART	0.95	0.85	0.90	0.82	0.55	0.69
	Avg.	**0.95**	**0.92**	**0.93**	**0.88**	**0.66**	**0.77**
Desc	T5/mT5	1.00	0.85	0.93	1.00	0.68	0.84
	GPT2/GPT2-pt	0.90	0.90	0.90	0.82	0.52	0.67
	BART/mBART	1.00	1.00	1.00	1.00	0.69	0.84
	Avg.	**0.97**	**0.92**	**0.94**	**0.94**	**0.63**	**0.78**
Kappa Statistic		**0.94**	**0.92**	**0.93**	**1.00**	**0.99**	**0.97**

Table 4: Results displaying the "Incorrect Number" error rates in English and Portuguese, categorized by strategies, with higher values indicating more errors. To facilitate comparison, we present results solely for the Monthly (DM) and Daily Deforestation (DD) domains, which are common to both languages. ∗(Lowest error rate among strategies averages) and †(Lowest error rate among model and strategy combinations).

varied depending on the language, strategy, and models used.

In English, the average results per strategy indicated that using text to represent numerical references did not yield a positive impact. This is evidenced by the `No Desc` strategy, which resulted in the lowest error rate. However, when examining the results per model, T5(`Desc Source`) strategy presented the lowest error rate, followed by BART(`Desc Source`) and T5(`No Desc`) strategies. In terms of automatic evaluation, the `Desc Target` strategy yielded the highest BLEU score with T5 being the best model in this strategy. The **Kappa** coefficient for inter-rater agreement regarding *Incorrect Number* error for both languages reached up to **0.90** according to Table 4, indicating a reasonable consensus between human evaluations.

Contrary to English, describing Portuguese numerical referring expressions in the `Desc Source` strategy resulted in the lowest error rate. The model with the fewest errors was GPT2-pt(`Desc Source`) strategy. Regarding the automatic evaluation, the `Desc` strategy yielded the highest BLEU score (0.68) with mT5, being the best model in this strategy for Portuguese.

It is important to note that Brazilian Portuguese approaches were evaluated across more domains than their English counterparts due to differences in both datasets. To compare the numerical error rate of models across languages, Table 4 presents the numerical error rate of approaches in daily and monthly Amazon deforestation domains, which share identical meaning representations in English and Portuguese. Based on the `Incorrect Number Error Rate` results, the `No Desc` was the best strategy in both languages. While error rates between daily and monthly deforestation were similar in English, Portuguese utterances in daily report format introduced fewer numerical errors than monthly reports, likely due to the higher amount of daily deforestation training sentences for Portuguese models.

6 Conclusion and Limitations

Finally, we revisit the research questions outlined in Section 1: **(RQ1)** A human evaluation was performed to annotate different error categories, such as numerical, named entities, context, word, uncheckable, other, and fluency errors. Results depicted across languages, models, and numerical strategies show the numerical error rate as the highest among the errors. Hence, concerning this research question, there is clear evidence that pure state-of-the-art large language models struggle to generate adequate and faithful numerical referring expressions. **(RQ2)** Results demonstrated that the Brazilian Portuguese approach `Desc Source` performs better. However, for English, representing numerical references in spell-out form did not help regardless of whether it was present in the source meaning representation (`Desc Source`), in the target text (`Desc Target`) or both (`Desc`). As depicted in Table 1, we report lower results for English when compared with Portuguese. This may result from the smaller size of the English dataset compared to Brazilian Portuguese. Moreover, surprisingly, for English, fine-tuning LLMs with smaller amounts of training data did not appear to produce higher results than originally hoped. More experiments will be needed however to verify this.

As evidenced in the results, this study confirms that Large Language Models struggle to generate numerical referring expressions, although T5 has performed better. The proposed strategy to solve the problem did not affect English, although it decreased numerical errors when describing the numbers on the source of Portuguese trials. Hence this strategy for describing numbers may help in low-resource scenarios.

For future work, we plan to extend our experiments to GPT3 and GPT4[6]. However, since these models are neither free, nor reproducible due to limited or no information concerning model size, architecture, training parameters, and data set creation, we will investigate related open-source variations such as BLOOM[7] and GPT-J[8].

7 Ethics Statement

As highlighted in the Human Evaluation Subsection 4.2, all annotators are members of the research group and were responsible for evaluating with an equal amount of occurrences; hence ethical approval for conducting research with human subjects was not required. All data is publicly available (see Data Subsection 2 for more information). No consent from data subjects was required as this data is purely factual, containing no personal data, and hence compliant with the EU's General Data Protection Regulation (GDPR)[9].

8 Acknowledgements

This publication has emanated from research conducted with the financial support of the National Council for Scientific and Technological Development (CNPQ) under grants 313103/2021-6 and 305753/2022-3; the Foundation for the Coordination and Improvement of Higher Education Personnel (CAPES) under grants 88887.488096/2020-00 and 88887.508597/2020-00; the State Funding Agency of Minas Gerais (FAPEMIG) under Grant No APQ-01.461-14; the Science Foundation Ireland under CRT-AI Grant No 18/CRT/622; and ADAPT, the SFI Research Centre for AI-Driven Digital Content Technology at Dublin City University under Grant No 13/RC/2106_P2. Furthermore, we thank the Center for Artificial Intelligence (C4AI-USP) and the support of the São Paulo Research Foundation (FAPESP Grant No 2019/07665-4) and IBM Corporation.

References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondřej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (WMT21). In *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88, Online. Association for Computational Linguistics.

João Campos, André Teixeira, Thiago Ferreira, Fábio Cozman, and Adriana Pagano. 2020. Towards fully automated news reporting in brazilian portuguese. In *Anais do XVII Encontro Nacional de Inteligência Artificial e Computacional*, pages 543–554. SBC.

Thiago Castro Ferreira, Diego Moussallem, Ákos Kádár, Sander Wubben, and Emiel Krahmer. 2018. NeuralREG: An end-to-end approach to referring expression generation. In *Proceedings of the 56th Annual*

[6] `https://openai.com/blog/chatgpt`

[7] BLOOM: BigScience Large Open-science Open-access Multilingual Language Model – `https://huggingface.co/bigscience/bloom`

[8] `https://huggingface.co/docs/transformers/model_doc/gptj`

[9] `https://gdpr-info.eu/recitals/no-159/`

Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1959–1969, Melbourne, Australia. Association for Computational Linguistics.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, et al. 2021. Finqa: A dataset of numerical reasoning over financial data. *arXiv preprint arXiv:2109.00122*.

Rossana Cunha, Thiago Castro Ferreira, Adriana Pagano, and Fabio Alves. 2020. Referring to what you know and do not know: Making referring expression generation models generalize to unseen entities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2261–2272, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ondřej Dušek and Filip Jurčíček. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 45–51, Berlin, Germany. Association for Computational Linguistics.

Sebastian Gehrmann, Falcon Dai, Henry Elder, and Alexander Rush. 2018. End-to-end content and plan selection for data-to-text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 46–56, Tilburg University, The Netherlands. Association for Computational Linguistics.

Pierre Guillou. 2020. Gportuguese-2 (portuguese gpt-2 small): a language model for portuguese text generation (and more nlp tasks...).

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Jeonghwan Kim, Junmo Kang, Kyung-min Kim, Giwon Hong, and Sung-Hyon Myaeng. 2022. Exploiting numerical-contextual knowledge to improve numerical reasoning in question answering. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1811–1821.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, Austin, Texas. 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.

Anna Björk Nikulásdóttir and Jón Guðnason. 2019. Bootstrapping a Text Normalization System for an Inflected Language. Numbers as a Test Case. In *Proc. Interspeech 2019*, pages 4455–4459.

OpenAI. 2023. Gpt-4 technical report.

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 on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Ratish Puduppully and Mirella Lapata. 2021. Data-to-text Generation with Macro Planning. *Transactions of the Association for Computational Linguistics*, 9:510–527.

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.

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

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

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Studies in Natural Language Processing. Cambridge University Press, Casmbridge, U.K.

Matthias C Rillig, Marlene Ågerstrand, Mohan Bi, Kenneth A Gould, and Uli Sauerland. 2023. Risks and benefits of large language models for the environment. *Environmental Science & Technology*, 57(9):3464–3466.

André Luiz Rosa Teixeira, João Campos, Rossana Cunha, Thiago Castro Ferreira, Adriana Pagano, and Fabio Cozman. 2020. DaMata: A robot-journalist covering the Brazilian Amazon deforestation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 103–106, Dublin, Ireland. Association for Computational Linguistics.

Richard Sproat. 2022. Boring problems are sometimes the most interesting. *Computational Linguistics*, 48(2):483–490.

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*.

Craig Thomson and Ehud Reiter. 2020. A gold standard methodology for evaluating accuracy in data-to-text systems. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 158–168, Dublin, Ireland. Association for Computational Linguistics.

Emiel van Miltenburg, Miruna Clinciu, Ondřej Dušek, Dimitra Gkatzia, Stephanie Inglis, Leo Leppänen, Saad Mahamood, Emma Manning, Stephanie Schoch, Craig Thomson, and Luou Wen. 2021. Underreporting of errors in NLG output, and what to do about it. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 140–153, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? Probing numeracy in embeddings. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pages 5307–5315.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721, Lisbon, Portugal. Association for Computational Linguistics.

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.

Hao Zhang, Richard Sproat, Axel H Ng, Felix Stahlberg, Xiaochang Peng, Kyle Gorman, and Brian Roark. 2019. Neural models of text normalization for speech applications. *Computational Linguistics*, 45(2):293–337.

A Appendix

A.1 Annotation Guidelines

After the most common error cases were identified and the treatment for the most difficult cases was agreed upon, annotators followed common guidelines for the rest of the evaluation process, as described in the following list:

- **Incorrect Number:** Has incorrect numerical values (e.g., model verbalizes an area value of “354” as “345”); Numerical values not verbalized in numerical form in the final texts were considered incorrect (e.g., “three hundred fifty-four” instead of “354”);
- **Incorrect Named Entity:** verbalizes entities incorrectly or verbalizes entities that do not exist;
- **Incorrect Word:** occurrence of spelling errors;
- **Context Error:** verbalizes some communicative intent incorrectly (e.g., verbalizes last month’s deforestation variation instead of total area deforestation);
- **Not checkable:** adds information that is not present in the input semantic representation in the verbalized text;
- **Other:** other types of verbalization errors;
- **Fluency:** the hypothesis verbalizes a not fluent text.

The annotation guidelines are summarised below:

- Entries were distributed in a collaborative spreadsheet.
- Each row consisted of the original Meaning Representation (MR), the generated hypothesis, and the rating categories.
- LLMs used to generate the entries were omitted in the spreadsheet.
- The spreadsheet was formatted to highlight the options (y - red; n - green) aiming to aid/ease the process with visual cues.
- Difficult cases were commented on to be further discussed within the group of raters, fostering improvements in the guidelines.
- Once, the annotation was finished, the spreadsheets were exported in .csv files for result computation.

Expected output The most affected state and municipality were respectively Pará (177.84 sq km) and Altamira, in the state of Pará (51.07 sq km). **Deforestation monthly intents** TOTAL_DEFORESTATION(area="177.84", location="deter-amz", month="4", state="PA", year="2021") [SEP] TOTAL_DEFORESTATION(area="51.07", city="Altamira", location="deter-amz", month="4", state="PA", year="2021")
T5 *nodesc* The most affected state and municipality were respectively Pará (**177.84 sq km**) and Altamira, in the state of Pará (**51.07 sq km**).
T5 *desctrg* The state with the most deforestation in the month was Pará (**one hundred and seventy-seven point eight four sq km**), and the most devastated municipality was Altamira / Pará, where deforestation amounted to **fifty-one point zero seven sq km**.
T5 *descsrc* The state with the most deforestation in the month was Pará (**177.84 sq km**), and the most devastated municipality was Altamira / Pará, where deforestation amounted to **51.07 sq km**.
T5 *desc* The state with the most deforestation in the month was Pará (**one hundred and seventy-seven point eight four sq km**), and the most devastated municipality was Altamira / Pará, where deforestation amounted to **fifty-one point zero seven sq km**.

Table 5: Sample from T5 outputs for English considering all 4 strategies. T5 performed as the best model for English. The numeric referring expressions are **bolded**.

A.2 Expected Output

A sample from the expected output is presented in Table 5 considering the meaning representation and each strategy in English. Furthermore, Tables 6 and 7 show Human Evaluation results for Portuguese and English languages and highlight problems regarding generating numerical referring expressions.

Language	Incorrect Number	Incorrect Named Entity	Incorrect Word	Context Error
Input	area="322.91"	city="Novo Progresso, Itaituba"	-	-
English	The National Institute for Space Research (INPE) estimated that deforestation of the Legal Amazon amounted to **2,322.91 sq km** in April two thousand and twenty, which is a one hundred and twenty-six percent increase from the previous month.	The National Institute for Space Research (INPE) reported that deforestation amounted to twenty-one point seven five sq km in the state of **Pará**, in the state of **Pará**, in February two thousand and twenty.	The main **class** of deforestation was clear-cut deforestation, which removes all **vegetetation** of the soil, responsible for 317.93 sq km of deforested area."	The most affected state and municipality were respectively **Pará** (177.84 sq km) and Altamira / Pará, in the state of Pará.
Input	cases="4091801" deaths="125584"	uc="PARQUE NACIONAL DO JAMANXIM"	-	-
Portuguese	São registrados, no total, **135.584** mortes e **4.093.801** casos de #COVID19 no Brasil.	O INPE gerou alerta para devastação (0,19 km²) causada pelo desmatamento com solo exposto, que remove totalmente a vegetação da floresta, no dia 10 de agosto de 2020 na **PARQUE NACIO**	A cidade mais atingida foi SANTAQUITÉRIA, **em CEARÁ**, que registrou 22 focos de incêndio.	O Instituto Nacional de Pesquisas Espaciais(INPE) registrou um total de quinhentos e sessenta e nove focos de queimadas no território brasileiro, no dia onze de outubro de dois mil e vinte, **o território brasileiro foi atingido**.

Table 6: Examples of categories of error in human evaluation for English and Brazilian Portuguese.

Language	Not Checkable	Other	Fluency Problem
Input	-	-	
English	The main cause of deforestation was the destruction of the soil, which leaves the soil clear of vegetation.	The National Institute for Space Research (INPE) in Pará, where the most affected municipality was **Novo Pro**	The National Institute for Space Research (INPE) reported that deforestation amounted to 21.75 sq km in the state of Pará, **in the state**
Input	area="0.32" day="22" month="8"	-	-
Portuguese	O INPE gerou alerta para devastação **(0,22 km2)** causada pelo desmatamento com solo exposto, que remove totalmente a vegetação da floresta, no dia 22 de agosto de 2020 na RESERVA EXTRATIVISTA VERDE PARA SEMPRE / Pará - no mês já são 2 dias com alertas e 0,32 km2 desmatar.	**A A A A A A A A BIOLÓGICA NASCENTES DA SERRA DO CACHIMBO** somou dois vírgula sete três km2 de área desmatada no mês de novembro de dois mil e vinte.	Com um total de mil quinhentos e setenta e oito vírgula oito sete km2, **o desmatamento com solo exposto**, deixando a terra sem vegetação, a principal causa de destruição da Amazônia Legal no mês foi o **desmatamento com solo exposto, deixando a terra sem vegetação**.

Table 7: Examples of categories of error in human evaluation for English and Brazilian Portuguese.

Using Locally Learnt Word Representations for better Textual Anomaly Detection

Alicia Breidenstein[1,2] and Matthieu Labeau[1]

[1]LTCI, Télécom-Paris, Institut Polytechnique de Paris, France
[2]Caisse des Dépôts et Consignations, Paris, France
{alicia.breidenstein, matthieu.labeau}@telecom-paris.fr

Abstract

The literature on general purpose textual Anomaly Detection is quite sparse, as most textual anomaly detection methods are implemented as out of domain detection in the context of pre-established classification tasks. Notably, in a field where pre-trained representations and models are of common use, the impact of the pre-training data on a task that lacks supervision has not been studied. In this paper, we use the simple setting of k-classes out anomaly detection and search for the best pairing of representation and classifier. We show that well-chosen embeddings allow a simple anomaly detection baseline such as OC-SVM to achieve similar results and even outperform deep state-of-the-art models.

1 Introduction

Anomaly Detection (AD) consists in detecting observations that deviate from *normality*: what is normal is defined by available data and assumed to be bounded (Ruff et al., 2021), while anomalies (which can be called *outliers*, or *novelty* depending on the application) are outside this bound. The most obvious hurdle with AD is that it is usually not possible to characterize anomalies: models are mostly not designed to target a specific type of outlier, and the assumptions made on data are rarely stated. In this context, supervision usually comes from *normal* data. However, most NLP models employ pre-trained representations: the impact that this kind of prior knowledge may have on AD is difficult to appreciate, and overlooked.

A first attempt to characterize outliers in natural language data was made by Arora et al. (2021), classifying them as coming from either *background* shifts (coming from a shift in domain) or *semantic* shifts (coming from a shift in content), and bringing insights on which detection method might better work on each. Arora et al. (2021) showed that background shifts are well detected by language models, which are able to estimate the density of normal data; there is furthermore an abundant literature on adapting pre-trained language model to new normal data (Ramponi and Plank, 2020).

We hence focus on semantic shifts, which are shown to be well detected by calibration methods. However, this assumes access to a classification model trained on relevant categories; we however prefer to not assume access to any labels, and adopt a simple but convenient way of evaluating AD: repurposing classification datasets by declaring one class to be normal and the others as anomalies, in what is called *k-classes-out*. In this setting, existing approaches are fewer. Some are inspired by topic modeling: they learn topic models optimized to reconstruct normal data well, aiming to detect anomalies by failing to accurately reconstruct them. For example, CVDD (Ruff et al., 2019) learns a limited number of topic-centroid vectors by applying attention upon pre-trained word-embeddings. A second direction is to train deep self-supervised models to recognize anomalies that are simulated, for example through random perturbation of data, as for DATE (Manolache et al., 2021). While both these models were previously compared on common datasets, CVDD uses pre-trained representations and DATE is only trained on the data available for the AD task.

In this paper, our goal is to investigate the impact of the pre-training data on anomaly detection performance in the k-classes-out setting; we experiment with static and contextual representations, off-the-shelf or obtained strictly on the AD training data, on three datasets. Our results show that the most simple configuration - a simple non-neural classification model, when equipped with textual representations obtained from the AD training data, can beat state-of-the-art models on our AD task.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 82–91
June 20, 2024

2 Background

2.1 Preliminaries

Anomaly score: To classify a data point $x \in \mathcal{X}$ as an anomaly, we compute an anomaly score $s : \mathcal{X} \rightarrow \mathbb{R}$ indicating its *degree of anomalousness* $s(x)$ (Ruff et al., 2021); then, a threshold δ is used as cutoff. However, we will here use measures that evaluate the performance of AD models using only s and independently of the choice of δ[1].

Data: We consider a training set of documents $\mathcal{D}_{train} = \{x_i\}_{i=1}^n$ for our task, which is part of a larger dataset: $\mathcal{D}_{train} \subset \mathcal{D}$. A document $x = (w_1, w_2, \ldots, w_l)$ is a sequence of $l \in \mathbb{N}$ words from a vocabulary $\mathcal{V}$. We will use different vector representations $\mathbf{x}$ of x depending on the method.

Pre-training word embeddings: In Ruff et al. (2019), CVDD is tested with embeddings $\mathbf{W} \in \mathbb{R}^{d\times|\mathcal{V}|}$ pre-trained with *FastText* and *GloVe*. However, those where trained on an external dataset, which might be very different than $\mathcal{D}_{train}$: hence, we propose to experiment with representations pre-trained on $\mathcal{D}_{train}$ and $\mathcal{D}$. We choose to use FastText, as the better performing static word representation algorithm. However, to avoid training prediction-based representations on datasets that are too small, we also use a traditional alternative in NLP, the *PPMI*(Church and Hanks, 1990) matrix, which we reduce to the appropriate dimension d using the SVD. As DATE is based on ELECTRA (Clark et al., 2020), we also experiment with representations obtained through its off-the-shelf version, and through one pre-trained on $\mathcal{D}$.

2.2 Anomaly Detection methods

We present in this section the necessary background information about the two models we experiment with, CVDD and DATE, as well as the chosen baseline. We follow Ruff et al. (2019) and use OC-SVM, a one-class classification-based AD model[2].

CVDD: CVDD scores a document by computing an average anomaly score over r *topics*. It takes as input word representations $\mathbf{X} = (\mathbf{w}_{w_j})_{j=1}^l \in \mathbb{R}^{d\times l}$. It learns jointly two components: (1) a multi-head self-attention mechanism, which computes sets of attention scores over the l input word embeddings for each of the r attention heads, grouped in $\mathbf{A} \in \mathbb{R}^{l\times r}$, allowing to aggregate them into r representations $\mathbf{M} = \mathbf{XA} \in \mathbb{R}^{d\times r}$, and (2) a set of r topic vectors $\mathbf{C} = (\mathbf{c}_k)_{k=1}^r \in \mathbb{R}^{d\times r}$ whose cosine distances with the corresponding training data representations $d(\mathbf{c}_k, \mathbf{m}_k)$ are minimized through the training objective. The anomaly score is, for a new document x_{test}, computed as follows:

$$s_{CVDD}(x_{test}) = \sum_{k=1}^{r} d(\mathbf{c}_k, \mathbf{X}_{test}\mathbf{a}_k)$$

DATE: DATE masks some of the tokens of the document, uses a generator to replace them, and learns through a transformer model D based on ELECTRA to detect the tokens which were modified, via a binary classification task called *Replaced Token Detection* (RTD). Motivated by computational efficiency, the authors propose to use as score the probability of each token *not being modified*:

$$s_{DATE}(x_{test}) = \frac{1}{l}\sum_{j=1}^{l} P_{RTD}(m_j = 0|x_{test}, D)$$

where m_j is a boolean indicating if the token w_j has been modified in the input to the model D. The model is trained to maximize the log-likelihood of this distribution on perturbed data. It is trained jointly using the *Replaced Mask Detection* (RMD) objective, which aims at predicting which masking pattern is used, and with the Masked Language Modeling (MLM) objective. DATE jointly learns its own contextual word representations, and is given the document x as input. It takes decisions at the token-level, which is made possible by using contextual representations. Note that the score s_{DATE} will give a high value to inliers examples, and should be reversed for comparison.

OC-SVM: We define our OC-SVM model following the baseline of CVDD: it uses the Scikit-learn (Pedregosa et al., 2011) implementation, based on the model described by Schölkopf et al. (2001). It takes as input the aggregate[3] $\mathbf{x}^{aggr} = \frac{1}{l}\sum_{j=1}^{l} \mathbf{w}_{w_j} \in \mathbb{R}^d$ and aims at separating all the training data points *from the origin* in the feature space $\mathcal{F}_k$. This space is defined as the reproducing

[1] Selecting this threshold is a difficult problem in itself, with values selected by validation not generalizing well (Khosla and Gangadharaiah, 2022).

[2] We also experimented on TONMF (Kannan et al., 2017) and their baseline LSA as well, but the results of these baselines were worse than the ones we obtain with CVDD, DATE and OC-SVM.

[3] Contrarily to Ruff et al. (2019), we don't present results using tf-idf to weight word embeddings, as we did not find it to produce competitive results.

kernel Hilbert space (RKHS) associated to the chosen positive semi-definite kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ and corresponding feature map $\phi_k : \mathbb{R}^d \to \mathcal{F}_k$. Separating data from the origin is done looking for a hyper-plane $\boldsymbol{\omega} \in \mathcal{F}_k$ maximizing a margin:

$$\min_{\boldsymbol{\omega},\rho,\boldsymbol{\xi}} \quad \frac{1}{2}\|\boldsymbol{\omega}\|^2 - \rho + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i$$
$$s.t \quad \rho - \langle \phi_k(\mathbf{x}_i^{aggr}), \boldsymbol{\omega}\rangle \leq \xi_i, \quad \xi_i \geq 0, \quad \forall i$$

where the margin to the origin is given by $\frac{\rho}{\|\boldsymbol{\omega}\|}$, and the $\boldsymbol{\xi}$ are the slack variables. The decision function should be positive for most training data points $\mathbf{x}_i^{aggr}$. Here, ν does not control the smoothness of the margin, but the *fraction of the data which the model will be allowed to consider as outliers*. Finally, the scoring function is simply minus the value given by f:

$$s_{OCSVM}(x_{test}) = \langle \phi_k(\mathbf{x}_{test}^{aggr}), \boldsymbol{\omega}\rangle - \rho$$

3 Experimental setting

We evaluate the performance of these models quantitatively on several datasets: after exploring the impact that the pre-training data used for word representations has on anomaly detection with OC-SVM and CVDD, we compare all models. [4]

3.1 Datasets

Following Manolache et al. (2021), we first compare the different methods on two publicly available textual datasets containing news articles for classification purposes: *20 Newsgroups*[5] and *AG News*[6]. The third dataset, *RNCP*[7], for *Répertoire National des Certifications Professionelles*, was built from a public official french repository with training certifications. The relevant statistics for the datasets are given in Table 5. For all datasets, we follow the pre-processing from Ruff et al. (2019).

3.2 Experimental details

Most of our experimental choices are made following Ruff et al. (2019). We mainly extend their experimental framework by looking at supplementary representations for the OC-SVM and CVDD models, trying to compare these approaches more fairly with respect to the data available to the model. Unless mentioned, for each model, we chose hyperparameters following the reference paper.

Evaluation with k-classes-out: Noting $\mathcal{C}$ the set of classes of the dataset $\mathcal{D}$, for each $c \in \mathcal{C}$ we have a train and test sets $\mathcal{D}^c_{train}$ and $\mathcal{D}^c_{test}$. In order to adapt the datasets to AD, one class c_{normal} is picked, while the others are considered to be anomalous. In our experimental setting, which we call *semi-supervised*, we consider that the normal class has been properly labeled, and the model is trained with exactly $\mathcal{D}^{c_{normal}}_{train}$. It is then evaluated on $\mathcal{D}_{test} = \bigcup_{c\in\mathcal{C}} \mathcal{D}^c_{test}$, where only elements of $\mathcal{D}^{c_{normal}}_{test}$ are to be recognized as inliers by the model. Experiments are repeated with taking every $c \in \mathcal{C}$ as c_{normal}. We present similar experiments in an *unsupervised* setting, where anomalies are present in the training data, in Appendix B.2.

Evaluation metrics: We use the Area Under Receiver Operating Curve (AUROC, or AUC) which is widely employed in the AD literature. It allows to measure the performance of a binary classifier by computing the area under the ROC curve, obtained by plotting the true positive rate against the false positive rate: hence, it covers the range of possible thresholds δ between normality and anomalies over the possible outputs of the anomaly score $s(x)$.

Experimenting with pre-trained representations: Following Section 2.1, we propose to experiment with various sets of representations for OC-SVM and CVDD: first, the FastText representations for English (and French, for RNCP) trained on Wikipedia and Common Crawl[8], which we note FT_{Large}. Then, we train our own embeddings with FastText on $\mathcal{D}$, and $\mathcal{D}^{c_{normal}}_{train}$, noting them respectively $\text{FT}_{\mathcal{D}}$ and $\text{FT}_{\mathcal{C}}$. Similarly, we note the representations obtained by reducing the dimension of a PPMI matrix[9] $\text{PPMI}_{\mathcal{D}}$ and $\text{PPMI}_{\mathcal{C}}$. For these pre-trained representations, we use $d = 300$. Lastly, we experimented with the ELECTRA model available on Huggingface[10] and one we trained on $\mathcal{D}$; as well as those obtained through the corresponding DATE model. As **none of the contextual representations gave competitive results**, we only display

[4] The code is available at https://github.com/abreidenstein/TextualAD

[5] http://qwone.com/~jason/20Newsgroups/

[6] http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

[7] https://www.data.gouv.fr/en/datasets/repertoire-national-des-certifications-professionnelles-et-repertoire-specifique/

[8] https://fasttext.cc/docs/en/crawl-vectors.html

[9] We follow here Turney (2012, Section 3.6) and don't use the eigenvalues when reducing the dimension.

[10] https://huggingface.co/google/electra-small-discriminator

20Ng	OC-SVM			CVDD	
	Linear	Poly	RBF	Best $r \in [1, 10]$	
$\text{FT}_\mathcal{D}$	**81.4** $\pm$ 0.1	**76.3** $\pm$ 0.1	58.4 $\pm$ 0.1	55.8 $\pm$ 0.3	$(r = 10)$
$\text{FT}_\mathcal{C}$	69.9 $\pm$ 0.2	69.7 $\pm$ 0.1	35.6 $\pm$ 0.1	50.0 $\pm$ 0.3	$(r = 5)$
FT_{Large}	66.0 $\pm$ 0.2	65.9 $\pm$ 0.2	66.4 $\pm$ 0.1	68.0 $\pm$ 0.1	$(r = 3)$
$\text{PPMI}_\mathcal{D}$	59.4 $\pm$ 0.2	59.1 $\pm$ 1.6	**75.1** $\pm$ 0.1	**70.4** $\pm$ 1.6	$(r = 2)$
$\text{PPMI}_\mathcal{C}$	74.5 $\pm$ 0.1	74.6 $\pm$ 0.1	40.6 $\pm$ 0.2	55.7 $\pm$ 0.2	$(r = 2)$

Table 1: AUCs of AD experiments over 20Ng, with OC-SVM with Linear, Poly and RBF kernels, and CVDD.

AGNews	OC-SVM			CVDD	
	Linear	Poly	RBF	Best $r \in [1, 10]$	
$\text{FT}_\mathcal{D}$	**89.8** $\pm$ 0.01	**87.7** $\pm$ 0.03	72.6 $\pm$ 0.1	86.5 $\pm$ 0.5	$(r = 1)$
$\text{FT}_\mathcal{C}$	79.6 $\pm$ 0.1	87.3 $\pm$ 0.1	20.8 $\pm$ 0.1	62.8 $\pm$ 0.6	$(r = 1)$
FT_{Large}	82.2 $\pm$ 0.1	82.0 $\pm$ 0.1	79.1 $\pm$ 0.1	**87.2** $\pm$ 0.7	$(r = 2)$
$\text{PPMI}_\mathcal{D}$	61.2 $\pm$ 0.1	60.6 $\pm$ 0.1	**89.4** $\pm$ 0.01	83.9 $\pm$ 0.2	$(r = 2)$
$\text{PPMI}_\mathcal{C}$	79.5 $\pm$ 0.1	79.8 $\pm$ 0.1	29.9 $\pm$ 0.1	58.7 $\pm$ 0.9	$(r = 5)$

Table 2: AUCs of AD experiments over AG News, with OC-SVM with Linear, Polynomial and RBF kernels, and CVDD.

RNCP	OC-SVM			CVDD	
	Linear	Poly	RBF	Best $r \in [1, 15]$	
$\text{FT}_\mathcal{D}$	**63.7** $\pm$ 0.05	**61.5** $\pm$ 0.04	**57.8** $\pm$ 0.05	**58.3** $\pm$ 0.4	$(r = 8)$
$\text{FT}_\mathcal{C}$	60.6 $\pm$ 0.04	60.8 $\pm$ 0.04	41.3 $\pm$ 0.1	52.2 $\pm$ 0.3	$(r = 10)$
FT_{Large}	56.2 $\pm$ 0.1	56.2 $\pm$ 0.2	55.0 $\pm$ 0.04	56.6 $\pm$ 0.3	$(r = 12)$
$\text{PPMI}_\mathcal{D}$	58.4 $\pm$ 0.04	58.6 $\pm$ 0.03	57.2 $\pm$ 0.1	56.9 $\pm$ 0.2	$(r = 2)$
$\text{PPMI}_\mathcal{C}$	57.4 $\pm$ 0.1	58.8 $\pm$ 0.1	49.0 $\pm$ 0.04	52.2 $\pm$ 0.1	$(r = 1)$

Table 3: AUCs of AD experiments over RNCP, with OC-SVM with Linear, Poly and RBF kernels, and CVDD.

the corresponding results in Appendix B.3.

4 Results

Choosing word representations: The results for CVDD and OC-SVM[11] obtained with the remaining static representations are presented in Table 1, 2 and 3 for two of the datasets. $\text{FT}_\mathcal{D}$ representations show consistently better performances than FT_{Large}, and the best overall, especially when used with an OC-SVM with a linear kernel. With class-based representations, the results of OC-SVM models seem to vary following the size of the dataset: the larger it is, the closer the results get to those of dataset-based representation. In particular, $\text{FT}_\mathcal{C}$ representations give great results on AG News with a polynomial kernel, as reported in Table 2. We hence postulate that the poorer performance of $\text{FT}_\mathcal{C}$ representations is linked to a lack of training data. With linear and polynomial kernels, $\text{PPMI}_\mathcal{C}$ give good results and largely beats $\text{PPMI}_\mathcal{D}$ on 20 Newsgroups and AG News, opposite to what we see with FT representations. We assume here that statistics obtained only on class data are more representative, and hence work better with simpler kernels. We discuss the poor performance of class-based representations with the RBF kernel in Appendix A.2.

[11]The scikit-learn implementation of OC-SVM is deterministic. Variations in our results come from the composition of document representations from word embeddings; we suppose this is due to how padding is handled in the implementation of (Ruff et al., 2019).

	AGNews	20Ng	RNCP
OC-SVM + FT_{Large}	82.2 $\pm$ 0.1	66.0 $\pm$ 0.2	56.2 $\pm$ 0.1
OC-SVM + $ours$	**89.8** $\pm$ 0.01	**81.4** $\pm$ 0.1	**63.7** $\pm$ 0.05
CVDD + FT_{Large}	87.2 $\pm$ 0.7	68.0 $\pm$ 0.1	56.6 $\pm$ 0.3
CVDD + $ours$	86.5 $\pm$ 0.5	70.4 $\pm$ 1.6	58.3 $\pm$ 0.4
DATE	*88.5* $\pm$ 0.2	*70.9* $\pm$ 0.4	*59.2* $\pm$ 0.1

Table 4: AUCs and standard deviations of AD experiments over all datasets, with all models. For OC-SVM and CVDD, we show the best results across hyperparameters with FT_{Large}, and across our own word representations, for which we took $\text{FT}_\mathcal{D}$ representations except for CVDD with 20 Newsgroups, where $\text{PPMI}_\mathcal{D}$ provide better results.

Overall comparison: Table 4 presents the best results obtained for each model, with comparison to DATE; additionally, for OC-SVM and CVDD, we present results for our representations (noted $ours$) and external representations separately. OC-SVM outperforms CVDD on all datasets. It reaches better results than DATE, especially on 20 Newsgroups and RNCP, although being far simpler. For all the models, the AUC values on the RNCP dataset are lower, which can be due to the shortness of the documents in this dataset, making the AD task more challenging.

On the performance of OC-SVM: our results show that, with appropriate representations, a simple OC-SVM model outmatches complex models such as CVDD and DATE. We hypothesize that, in our setting especially, AD approaches based on one-class classification are at an advantage; but the objective with which DATE is trained may lead the model away from what is needed in the k-classes-out setting, as it learns to detect random replacements. Here, the simplicity of an OC-SVM is a strength, though it has the disadvantage of not providing any density score nor possible word-level interpretation, contrarily to CVDD (through the attention mechanism) and DATE.

On the performance of dataset-based representations: our results show the clear superiority of representations pre-trained on the same data that will be used on the AD task. While dataset-based representations will generally not be available at training time, we argue that an OC-SVM model with class-based representations and a polynomial kernel should provide results that are very competitive with state-of-the-art models; the choice of representation pre-training method should depend on the quantity of training data available. Our results with contextual representations are in line with previous results from Ruff et al. (2019).

5 Conclusion

In this paper, we implement a fair comparison between existing textual anomaly detection methods in a k-classes-out setting and show that training the models on only the data available for the AD task can lead to better results. This allows methods regarded as baselines, such as OC-SVM models, to achieve impressive results, challenging state-of-the-art models based on deep neural architectures, with only the data available at hand. We intend to extend this line of work towards more challenging textual AD tasks. We also believe our results are indicative of the potential of model adaptation methods for semantic anomaly detection, which is a direction that has only been seldom explored (Xu et al., 2021). In the future, we also intend to extend our investigation to larger, more recent language models for obtaining representations.

6 References

Mira Ait-Saada and Mohamed Nadif. 2023. Unsupervised anomaly detection in multi-topic short-text corpora. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1392–1403, Dubrovnik, Croatia. Association for Computational Linguistics.

Udit Arora, William Huang, and He He. 2021. Types of out-of-distribution texts and how to detect them. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10687–10701, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

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*.

Songqiao Han, Xiyang Hu, Hailiang Huang, Mingqi Jiang, and Yue Zhao. 2022. Adbench: Anomaly detection benchmark.

Ramakrishnan Kannan, Hyenkyun Woo, Charu C. Aggarwal, and Haesun Park. 2017. *Outlier Detection for Text Data*.

Sopan Khosla and Rashmi Gangadharaiah. 2022. Evaluating the practical utility of confidence-score based techniques for unsupervised open-world classification. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*, pages 18–23, Dublin, Ireland. Association for Computational Linguistics.

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)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Nianzu Ma, Alexander Politowicz, Sahisnu Mazumder, Jiahua Chen, Bing Liu, Eric Robertson, and Scott Grigsby. 2021. Semantic novelty detection in natural language descriptions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 866–882, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Larry M. Manevitz and Malik Yousef. 2002. One-class svms for document classification. *J. Mach. Learn. Res.*, 2.

Andrei Manolache, Florin Brad, and Elena Burceanu. 2021. DATE: Detecting anomalies in text via self-supervision of transformers. In *Proceedings of the 2021 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 267–277, Online. Association for Computational Linguistics.

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.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Lukas Ruff, Jacob R. Kauffmann, Robert A. Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G. Dietterich, and Klaus-Robert Müller. 2021. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*.

Lukas Ruff, Yury Zemlyanskiy, Robert Vandermeulen, Thomas Schnake, and Marius Kloft. 2019. Self-attentive, multi-context one-class classification for unsupervised anomaly detection on text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4061–4071, Florence, Italy. Association for Computational Linguistics.

Bernhard Schölkopf, John Platt, John Shawe-Taylor, Alexander Smola, and Robert Williamson. 2001. Estimating support of a high-dimensional distribution. *Neural Computation*, 13:1443–1471.

Peter D Turney. 2012. Domain and function: A dual-space model of semantic relations and compositions. 44:533–585.

Keyang Xu, Tongzheng Ren, Shikun Zhang, Yihao Feng, and Caiming Xiong. 2021. Unsupervised out-of-domain detection via pre-trained transformers. 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 1052–1061, Online. Association for Computational Linguistics.

Hanlei Zhang, Xiaoteng Li, Hua Xu, Panpan Zhang, Kang Zhao, and Kai Gao. 2021. TEXTOIR: An integrated and visualized platform for text open intent recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 167–174, Online. Association for Computational Linguistics.

A Datasets and hyperparameters

A.1 Dataset description

Textual AD datasets and evaluation: To the best of our knowledge, only a handful of AD-specific textual datasets have been released: among them, CLINC150 (Larson et al., 2019), an intent classification[12] dataset comprising OOD examples, and the recent NSD2 (Ma et al., 2021) proposing anomalies that are created as fine-grained semantic modifications. As our objective is to get a clearer view of the performance of existing models, we choose to stay in the simple but popular setting of k-classes-out: we should note that this effectively restricts our study to the detection of what Arora et al. (2021) call semantic shifts. Many classification datasets have been used this way, a few of them being part of the recently released AD benchmark *ADBench* (Table B1: Han et al., 2022). Among those, we choose to re-use 20 Newsgroups and AG News, which DATE was applied to (Manolache et al., 2021). Following Ait-Saada and Nadif (2023), we diversify our experiments with a difficult classification dataset based on the French repository of training certifications, containing short texts (certification titles) with little lexical overlap within classes.

20 Newsgroups: This dataset is composed of newsgroups posts from 20 topics split between a training and a testing set. We reproduce the setup of Ruff et al. (2019); Manolache et al. (2021) and group the articles into 6 top-level categories.

AG News: This topic classification dataset was built by choosing the 4 largest classes from the original AG dataset and contains news articles collected from numerous news sources, and also includes an *train/test* split.

[12]Intent classification has attracted a large part of the efforts dedicated to textual AD, including a dedicated comparative framework (Zhang et al., 2021).

	$\lvert\mathcal{D}_{train}\rvert$			General statistics on $\mathcal{D}$			
Dataset	Smallest	Largest	Median	$\lvert\mathcal{D}_{train}\rvert/\lvert\mathcal{D}_{test}\rvert$ Ratio	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{V}\rvert$	Median(l)
AG News	30000	30000	30000	30000/1900	4	61230	24
20 newsgroups	577	2857	1916	0.6/0.4	6	76807	44
RNCP	927	14413	2957	0.75/0.25	16	4116	7

Table 5: Description of the datasets through key statistics.

RNCP: This dataset contains French training certification contents provided by the public organisation France Compétences. Following Ait-Saada and Nadif (2023), we build it into a classification dataset by taking as textual input the "Intitulé" (title) field, and using the ROME code of each certification (which are linked to thematic topics) for determining the class. However, their split into train and test sets was not made available: hence, while we keep the same $75\%/25\%$ ratio, we chose to work with an updated (and thus larger) version of the dataset.

A.2 Hyperparameter tuning

Hyperparameters and computation of results: Following Ruff et al. (2019), all presented values are obtained by averaging results over 5 runs. For OC-SVM, we present results over the best $\nu \in [0.05, 0.1, 0.2, 0.5]$. The best value of ν is then kept for experiments in section 4 and B.2. For CVDD, we only present the best results obtained over the number of attention heads r. Similarly, the best r are re-used in section section 4 and B.2. All our results are micro-averaged over all classes in the dataset, meaning that we average the values obtained for each model trained on $\mathcal{D}^c_{train}, \forall c \in \mathcal{C}$, weighted with $\lvert\mathcal{D}^c_{train}\rvert$. The standard deviation values presented are obtained using these averages over 5 different runs.

Choice of r for CVDD: Following (Ruff et al., 2019), we experiment with a large array of values for the number of context vectors r in CVDD. In our results, the best value seems to depend on both the dataset and the representation used, and needs to be tuned according to these two factors. The AUC variations given r on 20 Newsgroups for the 5 representations are presented in Figure 1. The best AUC values for FT$_{Large}$ and PPMI$_{\mathcal{D}}$ are obtained with $r = 3$ and $r = 2$ respectively. On the whole, the best values of r in Tables 2, 1, 3 show that more complex datasets lead CVDD to need more context vectors. Indeed, while the classes of AG News are thematically consistent, those of 20 Newsgroups aggregate several lower-level themes, and the documents in the RNCP classes are also quite diverse (Ait-Saada and Nadif, 2023).

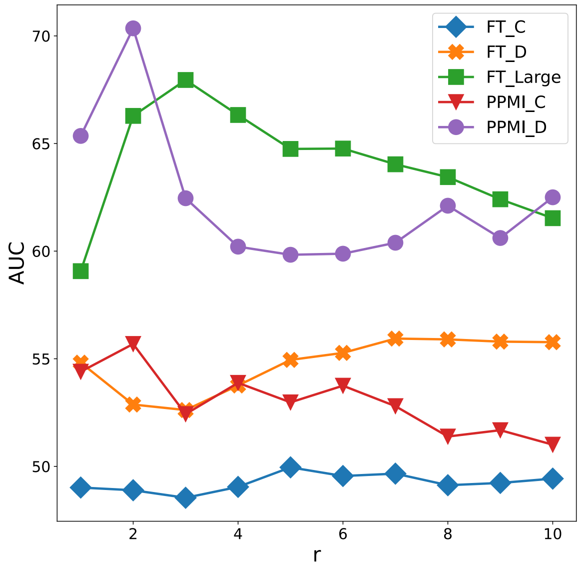


Figure 1: AUCs over 20 Newsgroups for CVDD models trained with our 5 pre-trained word embeddings (FT$_{\mathcal{C}}$, FT$_{\mathcal{D}}$, FT$_{Large}$, PPMI$_{\mathcal{C}}$, PPMI$_{\mathcal{D}}$), depending on the number of attention heads r.

A.3 Choice of OC-SVM Kernel

The RBF kernel is usually the default kernel for OC-SVMs (Manevitz and Yousef, 2002). However, a linear kernel provides here the best results. We can infer that the geometry of the FastText representations is well adapted to our AD task, and that using a more complex kernel makes the model prone to overfitting. In particular, following Ruff et al. (2019), we applied our hyperparameter search to ν only, whereas the γ hyperparameter of the RBF kernel is set automatically through a method proper to scikit-learn, inversely proportional to the variance of the training data. We hypothesize that the surprising counter performance of RBF kernel on class-based representations could be linked to this way of choosing γ. It may also be caused by examples very representative of normal data lying close to the origin in the feature space and being selected in the portion of training data ν allowed to be labelled by the OC-SVM as anomalies during training.

B Additional results

B.1 Evaluation metrics - AUPR

We also compare the performances of the different models using the Area Under Precision Recall curve (AUPR), which is less prevalent: it allows to measure the performance on imbalanced datasets, which is important in AD where the proportion of anomalies can be very low, although their detection matters the most. While this is not the case in our k-classes-out setting, we use this measure for complementary analysis.

Table 6 is an extended version of Table 4, which also includes values of AUPR-i and AUPR-o metrics, which are the AUPR values computed respectively for the inlier and the outlier classes. For this measure, the performances of a random classifier correspond to the number of positive examples divided by the size of the testing set. Thus, the results vary from one dataset to another, not only depending on the performances of the model, but also with the numbers of classes and their sizes. Overall the AUPR scores follow the same trend as the AUC score, except for CVDD which gets a better performance than the other models on the AUPR-i on 20 Newsgroups.

B.2 Unsupervised setting

Setting description: In our *unsupervised* setting, randomly selected documents of other classes are added to the normal class at a specified *contamination* rate r_{cont}. This corresponds to real-case scenarios where the data has not been properly labelled and may be contaminated with anomalies. More formally, this corresponds to using:

$$\mathcal{D}_{cont}^{c_{normal}} = \mathcal{D}_{train}^{c_{normal}} \cup \mathcal{D}_{anom}^{c_{normal}}$$

where $\mathcal{D}_{anom}^{c_{normal}}$ contains examples sampled from $\mathcal{D}_{train} \setminus \mathcal{D}_{train}^{c_{normal}}$ and r_{cont} is the proportion of these samples in $\mathcal{D}_{cont}^{c_{normal}}$. We experiment with several values of r_{cont} to evaluate the models robustness to anomalies in training data, and evaluate with the same $\mathcal{D}_{test}$. For fair comparison, we use the same contaminated datasets $\mathcal{D}_{cont}^{c_{normal}}$ for each model. Again, experiments are repeated by picking every $c \in \mathcal{C}$ to be c_{normal}.

Results: Figure 2 presents the results for several contamination rates r_{cont} corresponding to the proportion of anomalies added to the training set of the normal class $\mathcal{D}_{train}^{c_{normal}}$, for the three datasets. Unsurprisingly, the more the contamination rate rises, the lower the results get. We can notice that on 20 Newsgroups OC-SVM with a linear kernel seems less robust to anomalies in training data than the other methods. However, it still gives the best results. Overall, no particular trend stands out. While results obtained on RNCP decrease less with contamination, they are very unsatisfactory, for all models. We take note that specifically-designed methods based on a priori assumptions on the dataset reach better results (Ait-Saada and Nadif, 2023).

We should note that the results we obtain are, in some settings, notably worse than the ones presented in Manolache et al. (2021), especially on the dataset 20 Newsgroups, although we re-used the implementation provided by the authors and tried our best to reproduce their results following the paper. The discrepancy is particularly high for the OC-SVM and DATE models, while CVDD stays stable.

B.3 OC-SVM with DATE and Electra representations

To better understand the impact of local representations on AD, we experimented using the contextual representations from DATE with an OC-SVM model. These representations are learnt locally on each class of the dataset. To get a document-level representation, we used the $[CLS]$ token. We also experimented using Electra representations learnt locally without the additional RMD task present in DATE. Figure 7 presents the results on the different datasets.

On 20Ng and AGNews, combining DATE representations with OC-SVM shows worse performances than the ones obtained by DATE (with DATE representations) or OC-SVM (with FastText representations) in Table 4. On the RNCP Dataset however, using OC-SVM with DATE representations gets the best results. We hypothesize that the shortness of RNCP documents leads smaller models such as FastText to have more difficulties to extract the relevant information in the representations. However, AD methods specifically designed for short text documents such as the one presented by Ait-Saada and Nadif (2023) still provide the best results.

	AGNews			20Ng			RNCP		
	AUC	AUPR-i	AUPR-o	AUC	AUPR-i	AUPR-o	AUC	AUPR-i	AUPR-o
OC-SVM + FT$_{Large}$	82.2 ± 0.1	68.1 ± 0.2	90.8 ± 0.04	66.0 ± 0.2	34.8 ± 0.2	86.8 ± 0.1	56.2 ± 0.1	12.6 ± 0.1	91.7 ± 0.04
OC-SVM + *ours*	**89.8** ± 0.01	**75.7** ± 0.1	**96.0** ± 0.01	**81.4** ± 0.1	*44.3* ± 0.2	**94.4** ± 0.1	**63.7** ± 0.05	**14.5** ± 0.03	**93.2** ± 0.01
CVDD + FT$_{Large}$	87.2 ± 0.7	71.6 ± 0.8	94.4 ± 0.4	68.0 ± 0.1	42.5 ± 0.2	86.6 ± 0.03	56.6 ± 0.3	12.8 ± 0.2	91.5 ± 0.1
CVDD + *ours*	86.5 ± 0.5	70.3 ± 1.1	94.2 ± 0.2	70.4 ± 1.6	**45.3** ± 1.8	88.2 ± 0.4	58.3 ± 0.4	12.8 ± 0.2	91.8 ± 0.1
DATE	*88.5* ± 0.2	*73.7* ± 0.6	*95.2* ± 0.1	*70.9* ± 0.4	41.8 ± 0.5	*89.8* ± 0.1	*59.2* ± 0.1	*13.1* ± 0.1	*92.6* ± 0.04

Table 6: AUCs of AD experiments over all datasets, with all models. For OC-SVM and CVDD, we show the best results across hyperparameters with FT$_{Large}$, and across our own word representations.

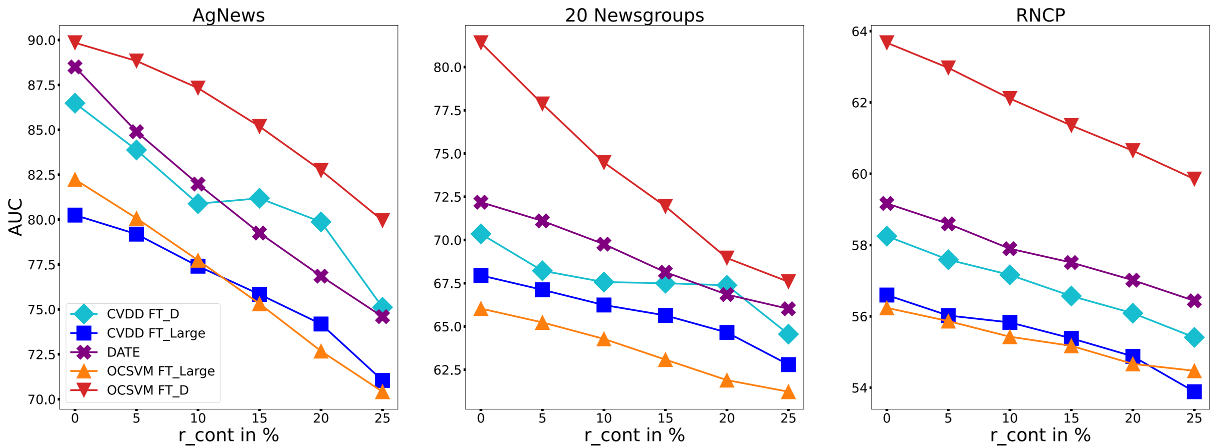


Figure 2: AUCs of AD experiments over AG News, 20 Newsgroups and RNCP, with 5 of the models shown in Table 6, for a contamination rate r_{cont} varying from 0 to 25%.

		AGNews	20Ng	RNCP
DATE	OC-SVM Linear	73.1	63.3	65.5
	OC-SVM Poly	73.3	63.2	67.7
	OC-SVM Rbf	73.2	63.8	66.7
Electra	OC-SVM Linear	41.5	61.4	59.2
	OC-SVM Poly	40.9	61.3	59.3
	OC-SVM Rbf	40.4	61.4	59.3

Table 7: AUCs of AD experiments over all datasets, with OC-SVM using representations from DATE and Electra learnt on each class of the dataset.

Using locally trained Electra representations combined with OC-SVM gets worse results than using DATE representations. This underlines the contribution of the RMD task introduced by Manolache et al. (2021) for AD. We also experimented on OC-SVM with pre-trained Electra embeddings, but got notably worse results than the ones presented in Table 7. We recall that Ruff et al. (2019) also experimented with BERT representations but found the results to be lacking and did not display them.

B.4 Results detailed by class

20 Ng	**OC-SVM**		**CVDD**		**DATE**
Class	linear - $\text{FT}_{\mathcal{D}}$	linear - FT_{Large}	$\text{FT}_{\mathcal{D}}$ $(r = 2)$	FT_{Large} $(r = 3)$	DATE
$0 - comp$	87.4	78.9	78.0	73.7	87.7
$1 - misc$	86.5	63.8	65.1	74.0	54.8
$2 - pol$	82.7	58.6	76.3	71.4	61.3
$3 - rec$	82.4	63.4	69.1	60.5	66.9
$4 - rel$	83.1	67.2	75.8	77.9	71.0
$5 - sci$	69.4	57.2	55.8	58.2	64.5

Table 8: AUCs of AD experiments over the different classes of 20 Newsgroups dataset, with all models. For OC-SVM and CVDD, we show the best results across hyperparameters with FT_{Large}, and across our own word representations.

AG News	**OC-SVM**		**CVDD**		**DATE**
Class	linear - $\text{FT}_{\mathcal{D}}$	linear - FT_{Large}	$\text{FT}_{\mathcal{D}}$ $(r = 2)$	FT_{Large} $(r = 3)$	DATE
$0 - business$	85.2	77.8	83.9	87.9	88.7
$1 - science$	86.3	74.8	80.7	83.4	82.6
$2 - sports$	95.7	92.1	94.7	95.7	94.5
$3 - world$	92.1	83.3	86.5	81.8	88.2

Table 9: AUCs of AD experiments over the different classes of AG News dataset, with all models. For OC-SVM and CVDD, we show the best results across hyperparameters with FT_{Large}, and across our own word representations.

RNCP	**OC-SVM**		**CVDD**		**DATE**
Class	linear - $\text{FT}_{\mathcal{D}}$	linear - FT_{Large}	$\text{FT}_{\mathcal{D}}$ $(r = 8)$	FT_{Large} $(r = 12)$	DATE
$1 - environnement$	52.7 6	50.2	53.5	52.8	55.1
$2 - defense$	73.7	51.9	66.2	59.4	38.7
$3 - patrimoine$	63.1	47.5	59.3	55.4	63.5
$4 - economie$	58.7	56.6	53.0	53.8	55.6
$5 - recherche$	65.7	58.4	65.7	65.1	66.5
$6 - nautisme$	57.1	50.9	55.7	54.1	57.7
$7 - aronautique$	68.7	63.5	63.7	62.7	66.4
$8 - scurit$	72.3	65.4	72.1	74.3	57.2
$9 - multimdia$	71.7	62.1	57.6	56.0	60.2
$10 - humanitaire$	61.3	51.8	56.8	54.6	58.3
$11 - nuclaire$	69.4	63.2	62.7	61.2	63.1
$12 - enfance$	81.4	55.5	67.5	56.3	61.7
$13 - saisonnier$	76.7	51.5	70.6	54.8	44.0
$14 - assistance$	65.5	41.5	49.7	38.7	50.1
$15 - sport$	68.1	51.2	56.9	48.3	58.2
$16 - ingnierie$	67.8	62.7	62.2	63.3	65.6

Table 10: AUCs of AD experiments over the different classes of RNCP dataset, with all models. For OC-SVM and CVDD, we show the best results across hyperparameters with FT_{Large}, and across our own word representations.

Can probing classifiers reveal the learning by contact center large language models?: *No, it doesn't!*

Varun Nathan, Ayush Kumar and **Digvijay Ingle**
{varun.nathan, ayush, digvijay.ingle}@observe.ai
Observe.AI
Bangalore, India

Abstract

Fine-tuning large language models (LLMs) with domain-specific instruction dataset has emerged as an effective method to enhance their domain-specific understanding. Yet, there is limited work that examines the core characteristics acquired during this process. In this study, we benchmark the fundamental characteristics learned by contact-center (CC) domain specific instruction fine-tuned LLMs with out-of-the-box (OOB) LLMs via probing tasks encompassing conversational, channel, and automatic speech recognition (ASR) properties. We explore different LLM architectures (Flan-T5 and Llama) and sizes (3B, 7B, 11B, 13B). Our findings reveal remarkable effectiveness of CC-LLMs on the in-domain downstream tasks, with improvement in response acceptability by over 48% compared to OOB-LLMs. However, we observe that the performance of probing classifiers are relatively similar and does not reflect the performance of in-domain downstream tasks. A similar observation is also noted on SentEval dataset that assess capabilities of models in terms of surface, syntactic, and semantic information through probing tasks. Our study challenges the premise that probing classifiers can reveal the fundamental characteristics learned by large language models and is reflective of the downstream task performance, via a case-study of LLMs tuned for contact center domain.

1 Introduction and Related Works

Large Language models (LLMs) have made significant strides in recent years, with their ability to generate fluent text on variety of inputs (Wei et al., 2022; OpenAI, 2023). The strategy of fine-tuning the general-purpose models with domain-specific data has led to performance improvements in domains with LLMs such as BioGPT (Luo et al., 2022) and Med-PaLM (Singhal et al., 2023) in biomedical research, CodeT5 (Wang et al., 2021), CodeLLaMa in coding (Rozière et al., 2023), and Bloomberg-GPT (Wu et al., 2023) in finance, demonstrating the need and advantage of domain specific fine-tuning of LLMs. However, one domain that has received relatively little attention is the contact center industry. Contact centers play a crucial role in customer service and support for various businesses. They address a broad spectrum of customer queries, from technical issues to billing concerns. Incorporating LLMs into contact center workflows have a potential to transform the sector. However, noisy queries, spontaneous conversational dynamics and domain specific understanding pose significant challenges for LLMs. Adapting to these nuances is crucial for LLMs to enhance their effectiveness in contact center.

Instruction fine-tuning (Longpre et al., 2023) has emerged as one the promising approaches to develop domain-specific LLMs. Assessing effectiveness of LLMs often involves evaluating their performance on specific downstream tasks. However, probing the representations of the models on different probing tasks provide a deeper insight into the fundamental aspects of what language models capture and learn (Conneau et al., 2018). These tasks have been instrumental in understanding the underlying characteristic of language models. Conneau et al. (2018) introduced probing tasks in SentEval to assess sentence embedding representations of language models. Following this, studies like those by Tenney et al. (2019) and Lin et al. (2019) have applied layer-wise probing to BERT, shedding light on its semantic and hierarchical processing capabilities. While the majority of probing studies have concentrated on general LMs, work by Kumar et al. (2021) delved into the representation capabilities of RoBERTa in contact center domain. Building on this foundation, our study seeks to further understand the intricacies of instruction-fine-tuned LLMs in contact centers through specific research questions, aiming to uncover how these LLMs adapt and learn within this specialized context:

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 92–100
June 20, 2024 ©2024 Association for Computational Linguistics

Figure 1: Quality of responses generated by CC LLMs versus OOB LLMs on downstream tasks in contact-center domain using a scale of *Extremely Bad* response to *Extremely Good* response. We note that CC LLMs result in over 48% improvement in response acceptability (>=*Acceptable*) compared to OOB LLMs (Flan-T5, Llama-Instruct).

- **RQ1:** How effective is instruction fine-tuning in enhancing LLMs' performance on downstream tasks within the contact-center domain?
- **RQ2:** What unique properties related to contact-center interactions are acquired by LLMs fine-tuned on CC instruction sets, compared to out-of-the-box models?
- **RQ3:** How does the choice of model architecture and size influence LLMs' performance on probing tasks?
- **RQ4:** Following domain-specific instruction fine-tuning, what general-purpose fundamental properties do LLMs retain?

2 Training Contact-Center LLM

In this work, we train a contact center-specific large language model (CC-LLM) using a proprietary dataset of ASR transcripts[1] from various sectors. Through instruction fine-tuning, we adapt out-of-box (OOB) LLMs to the contact center conversations, characterized by multi-party interactions, disfluencies, and ASR errors. Our training methodology involves generating diverse instructions for a wide array of tasks, such as call summarization, dialog question answering etc., to tailor the model's capabilities for contact center applications. More details is mentioned in Section A.1.

3 Probing tasks

Probing tasks tailored to the contact center domain provide valuable insights into the capabilities and limitations of LMs in this specific area, as demonstrated in a previous study (Kumar et al., 2021). In their work, the authors propose probing tasks to investigate the conversational, channel, and ASR properties of pre-trained LMs. We refer to these probing tasks and utilize the details outlined in the work to construct datasets to investigate the characteristics of contact-center LLMs via the probing tasks. Additionally, we also probe the LMs on a benchmark probing task of SentEval suite (Conneau et al., 2018) that aims to uncover the linguistic knowledge and underlying properties learned by the model. SentEval suite consists of probing tasks across the categories of surface information, syntactic information and semantic information.

4 Implementation Details

We compare two model classes, namely Flan (Longpre et al., 2023) and Llama (Touvron et al., 2023) in the three categories: OOB foundation model, OOB instruction model, and the CC instruction model. Following the previous work by Alain and Bengio, 2017, we utilize one-layer linear MLP classifier to train probing classifiers on the representations extracted from LLMs on the concatenated input of *{task-instruction, dialog/turn transcript}*. More details is outlined in Section A.2.

5 Results and Analysis

5.1 RQ1: Performance on downstream tasks

We perform a qualitative assessment of the responses generated by CC and OOB-LLMs by cate-

[1] We cannot release the dataset due to proprietary reasons.

Probing Tasks	OOB Foundation				OOB Instruction-Tuned				Contact Center			
	OOB-T5 (3B)	OOB-T5 (11B)	OOB-Llama (7B)	OOB-Llama (13B)	OOB-Flan-T5 (3B)	OOB-Flan-T5 (11B)	OOB-Llama-Instruct (7B)	OOB-Llama-Instruct (13B)	CC-Flan-T5 (3B)	CC-Flan-T5 (11B)	CC-Llama-Instruct (7B)	CC-Llama-Instruct (11B)
Disfluency	72.12	71.97	68.30	71.57	71.72	73.03	68.88	69.81	72.24	72.89	69.16	67.83
Pause	80.90	80.70	77.79	81.25	82.09	83.45	80.45	80.25	81.78	83.00	76.85	79.24
Overtalk	86.95	89.55	82.79	81.59	89.45	90.70	83.25	77.70	87.80	88.19	72.55	78.92
Question	77.52	74.49	70.34	74.31	77.59	75.39	73.03	74.33	76.96	80.37	76.22	77.15
Speaker	80.95	81.96	77.54	82.70	82.55	83.39	80.26	80.21	82.11	82.72	78.70	79.94
Resp. Length	67.65	69.35	66.23	69.09	69.20	69.66	66.03	67.29	68.88	68.79	67.27	67.95
Turn Taking	68.30	69.14	65.01	69.33	64.30	67.66	69.62	68.65	66.83	69.59	62.50	63.45
Token Multi	52.45	49.32	40.71	42.64	59.91	63.07	43.02	40.60	59.31	60.73	41.62	42.85
Token Binary	60.50	60.48	50.07	54.93	68.34	73.12	49.84	48.77	70.11	70.07	49.88	50.14
Avg. Score	71.93	71.88	66.53	69.71	73.90	75.50	68.26	67.51	74.00	75.15	66.08	67.50

Table 1: Benchmarking CC and OOB LLMs in terms of Macro F1 evaluated on contact-center probing tasks.

gorizing the responses generated by each of them into one among following seven classes: *Extremely Good*, *Very Good*, *Good*, *Acceptable*, *Bad*, *Very Bad*, and *Extremely Bad*. The annotation process in detail is mentioned in Section A.3. We analyze the responses generated by the LLM groups, and observe significant difference in the distribution of quality of responses (refer Figure 1). Specifically, responses generated by OOB-T5 (11B) (Raffel et al., 2020), OOB-Flan-T5 (11B), OOB-Llama (13B) and OOB-Llama-Instruct (13B) models are consistently skewed towards the lower end of the quality spectrum. A majority of these responses fall within the *Bad* to *Extremely Bad* categories, indicating that without specific fine-tuning, OOB models struggle to generate satisfactory responses for contact center specific instructions. Conversely, responses generated by CC-Flan-T5 (11B) and CC-Llama (13B) models exhibit a notable shift towards higher quality categories. A substantial portion of responses generated by these models lands in the *Acceptable* to *Extremely Good* range, demonstrating their ability to comprehend and generate contextually relevant responses for contact center interactions. Specifically, 91% of responses from CC-Flan-T5 and 87% of responses from CC-Llama has score >=*Acceptable* compared to 22% and 39% from respective OOB instruction models. This improvement in performance can be attributed to the fine-tuning process with contact center data.

5.2 RQ2: Contact-center probing tasks

In order to investigate the conversational properties learnt by CC-LLMs that lead to performance superior to OOB-LLMs, we evaluate these models on the probing tasks in Section 3 and per the methodology described in Section A.2. Although our probing tasks are carefully designed to uncover the latent knowledge within these models, our findings in Table 1 did not conclusively favor either type of LLM. Specifically, we observe a mixed trend where 1 out of 4 CC models, CC-Flan-T5 (3B) have higher average score and 2 out of 4 models, CC-Flan-T5 (11B) and CC-Llama (13B), have marginally lower ($< 0.5\%$) average score compared to their corresponding OOB instruction-tuned counterparts. We also note a similar observation when comparing CC-LLMs with OOB foundation models wherein 3 out of 4 CC-LLMs have comparable or better average score. This intriguing result prompts us to delve deeper into several critical aspects of LLMs and their fine-tuning process prompting us to put forth following opportunities for exploration. **Probing via Hidden Layer Representation:** While this method has been widely employed (Kumar et al., 2021; Fayyaz et al., 2021; Thukral et al., 2021) to unearth linguistic properties by language models, we question whether it is sufficiently nuanced to capture conversational intricacies. It is conceivable that the differences we seek are not embedded in the representations extracted but are instead contingent on the decoding strategy employed during the language generation process. This insight underscores the pivotal role of decoding strategies in converting latent embeddings into coherent sequences of tokens that reflect both the given instruction and input. It prompts us to consider that instructing and fine-tuning a general-purpose model and a domain-specific model may ultimately hinge on decoding proficiency rather than vastly divergent learned representations. We believe that this calls for a deeper investigation into designing right

Probing Tasks	OOB Foundation		OOB Instruction Tuned		Contact Center	
	OOB-T5 (11B)	OOB-Llama (13B)	OOB-Flan-T5 (11B)	OOB-Llama-Instruct (13B)	CC-Flan-T5 (11B)	CC-Llama-Instruct (13B)
Bigram Shift	92.48	85.66	94.19	85.59	92.17	76.79
Coordination Inversion	79.36	68.65	77.59	71.68	76.59	70.30
Object Number	82.70	73.90	89.20	74.20	86.90	76.49
Odd Man Out	73.69	66.09	74.99	66.90	72.99	63.51
Past Present	88.99	84.17	89.19	85.19	89.59	82.98
Sentence Length	100.00	100.00	100.00	100.00	100.00	100.00
Subj Number	86.19	79.49	92.09	79.66	90.29	81.57
Top Constituents	68.85	73.98	74.65	67.44	75.78	58.55
Tree Depth	36.02	28.73	37.24	32.65	38.49	27.92
Average Score	78.70	73.41	81.02	73.70	80.31	70.90

Table 2: Benchmarking CC and OOB LLMs in terms of Macro F1 evaluated on SentEval probing tasks.

probing strategies for recently popular generative language models trained via instruction fine-tuning.

Re-designing probing tasks: The existing set of probing tasks, although comprehensive, may not fully encapsulate the diverse landscape of conversational properties. Conversations are inherently dynamic, context-dependent, and influenced by various factors, including the interplay between participants, the history of the conversation, long-context dependencies and the evolution of topics. However the probing tasks in Kumar et al. (2021) are designed for single utterance inputs. Such scenario may not fully capture these dynamic aspects of conversation. It is plausible that more specific probing tasks tailored to the characteristic of contact center interactions are needed to fully conclude the learnings of the LLMs. These tasks should ideally mirror the challenges posed by real-world downstream applications that help diagnose the contextual properties and the interplay in the conversations.

5.3 RQ3: Model architecture and model size

From our results in Table 1, we note that T5 models consistently outperform Llama models across the three settings, OOB Foundation, OOB Instruction-tuned and Contact Center, highlighting that T5's encoder-decoder architecture has better learnt to comprehend conversational properties compared to Llama's decoder only architecture. Similarly, in downstream task performance (Section 5.1), CC-Flan-T5 (11B), although smaller in size, outperforms CC-Llama (13B). This outcome was surprising, especially considering Flan's smaller size and the Llama model's widespread popularity in the open-source community. It leads to question the impact of model architecture versus size in accurately comprehending the conversational contexts.

5.4 RQ4: General purpose probing tasks

Post fine-tuning on contact center instruction data, CC-Flan-T5 and CC-Llama show a reduced dependency on fundamental linguistic properties as evidenced by the decreased average score on SentEval probing suite. Consistent with prior findings, the Llama models exhibits a lower score compared to Flan models on general purpose probing task as well. Additionally, we note that while performance of CC-Flan-T5 is lower than OOB-Flan-T5 by 0.7%, this drop is 2.8% in Llama. This again suggests distinct learning mechanisms between encoder-decoder and decoder-only architectures, warranting further investigation in the community.

6 Conclusion

Our study contributes to the growing body of research on fine-tuning LLMs with domain-specific instructions. In this work, we demonstrate that CC-LLMs, CC-Flan-T5 and CC-Llama, exhibit superior performance on downstream tasks within the contact center domain. This finding reinforces the effectiveness of fine-tuning LLMs with domain-specific instructions, as expected. However, our comparison between OOB and CC models on the probing task reveals intriguing and unexpected observations. While the performance of CC-LLMs are much superior to the OOB-LLMs on downstream tasks, the performance of probing classifiers across the models shows no substantial differences. This questions the efficacy of traditional probing mechanisms and probing tasks in understanding the LLMs. We also observe that the decoder model (Llama-13B) consistently underperforms compared to the lower sized encoder-decoder model (Flan-11B) in all experiments This prompts more research into the learning dynamics of these architectures.

Limitations

While our study provides valuable insights into training a contact-center specific language model and conducting linear edge probing, it is important to acknowledge certain limitations in our work. Firstly, our exploration of language models is limited to a couple of models belonging to two architectures, one encoder-decoder and one decoder style. We choose these models on the basis of their effectiveness across different tasks as has been surfaced up in the research community, however, the trends we observe may not necessarily hold true for other models within the same class of architecture. Secondly, our work is based on the probing methodology of linear edge probing, which applies a one layer linear MLP on hidden representations. The performance and observations on probing tasks may differ if a different probing setup, such as an attention-based probing, is used. It is crucial to explore alternative probing methods to gain a more comprehensive understanding of the language model's characteristics. Moreover, the set of probing tasks we utilize may not cover the full range of characteristics that a language model can encode. Additional probing tasks can be considered to do a more extensive study of the model's capabilities. Lastly, our research is conducted on a proprietary dataset that cannot be released. This limits the ability of other researchers to directly compare their results or replicate our experiments. Access to the dataset is crucial for future work in this area, and we encourage the development of publicly available datasets for domain-specific language models.

Despite these limitations, our study underscores the importance of domain-specific instruction models and highlights the limited capacity of general-purpose language models to meet domain specific use-cases. Furthermore, we pose thought-provoking questions that can guide further research and contribute to the advancement of the research community's understanding of the properties encoded in generative language models in the new era.

References

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Afra Amini and Massimiliano Ciaramita. 2023. Probing in context: Toward building robust classifiers via probing large language models. *CoRR*, abs/2305.14171.

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. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Alexis Conneau, Germán 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, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2126–2136. Association for Computational Linguistics.

Mohsen Fayyaz, Ehsan Aghazadeh, Ali Modarressi, Hosein Mohebbi, and Mohammad Taher Pilehvar. 2021. Not all models localize linguistic knowledge in the same place: A layer-wise probing on bertoids' representations. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2021, Punta Cana, Dominican Republic, November 11, 2021*, pages 375–388. Association for Computational Linguistics.

Ayush Kumar, Mukuntha Narayanan Sundararaman, and Jithendra Vepa. 2021. What BERT based language model learns in spoken transcripts: An empirical study. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2021, Punta Cana, Dominican Republic, November 11, 2021*, pages 322–336. Association for Computational Linguistics.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Mishig Davaadorj, Joel Lamy-Poirier, João Monteiro, Oleh Shliazhko, Nicolas Gontier, Nicholas Meade, Armel Zebaze, Ming-Ho Yee, Logesh Kumar Umapathi, Jian Zhu, Benjamin Lipkin, Muhtasham Oblokulov, Zhiruo Wang, Rudra Murthy V, Jason Stillerman, Siva Sankalp Patel, Dmitry Abulkhanov, Marco Zocca, Manan Dey, Zhihan Zhang, Nour Moustafa-Fahmy, Urvashi Bhattacharyya, Wenhao Yu, Swayam Singh, Sasha Luccioni, Paulo Villegas, Maxim Kunakov, Fedor Zhdanov, Manuel Romero, Tony Lee, Nadav Timor, Jennifer Ding, Claire Schlesinger, Hailey Schoelkopf, Jan Ebert, Tri Dao, Mayank Mishra, Alex Gu, Jennifer Robinson, Carolyn Jane Anderson, Brendan Dolan-Gavitt, Danish Contractor, Siva Reddy, Daniel Fried, Dzmitry Bahdanau, Yacine Jernite, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. 2023. Starcoder: may the source be with you! *CoRR*, abs/2305.06161.

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, BlackboxNLP@ACL 2019, Florence, Italy, August 1, 2019*, pages 241–253. Association for Computational Linguistics.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 22631–22648. PMLR.

Renqian Luo, Liai Sun, Yingce Xia, Tao Qin, Sheng Zhang, Hoifung Poon, and Tie-Yan Liu. 2022. Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Briefings Bioinform.*, 23(6).

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2023. Code llama: Open foundation models for code. *CoRR*, abs/2308.12950.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. 2023. Large language models encode clinical knowledge. *Nature*, pages 1–9.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *CoRR*, abs/2211.09085.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. 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 4593–4601. Association for Computational Linguistics.

Shivin Thukral, Kunal Kukreja, and Christian Kavouras. 2021. Probing language models for understanding of temporal expressions. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2021, Punta Cana, Dominican Republic, November 11, 2021*, pages 396–406. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Yue Wang, Weishi Wang, Shafiq R. Joty, and Steven C. H. Hoi. 2021. Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8696–8708. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y. 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 *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David S. Rosenberg, and Gideon Mann. 2023. Bloomberggpt: A large language model for finance. *CoRR*, abs/2303.17564.

A Appendix

A.1 Training Contact-Center Instruction-Tuned LLM

Numerous closed-source (Brown et al., 2020; OpenAI, 2023) and open-source (Touvron et al., 2023) general purpose LLMs have demonstrated abilities to address a diverse range of tasks in natural language processing. However, specialised models like CodeT5 (Wang et al., 2021), StarCoder (Li et al., 2023), Med-PaLM (Singhal et al., 2023), BioGPT (Luo et al., 2022), Galactica (Taylor et al., 2022), BloombergGPT (Wu et al., 2023) emphasize the significance of domain-specific models in achieving exceptional performance within fields like coding, bio-medicine, science, and finance. These models excel at producing high-quality outputs and tackling domain-specific challenges, illustrating the need of tailored LMs in diverse domains.

Inspired by the above works, we leverage inhouse dataset[2] of conversational interactions between agents and customers to train a CC-specific LLM (CC-LLM) to model the properties of CC conversations. Due to the spontaneous nature of these conversations, the data is often nuanced with characteristics such as multi-party speakers, disfluencies, overtalks, call transfers, etc. Furthermore, the data is obtained post transcription from an automatic speech recognition (ASR) system, thus introducing the challenge of dealing with ASR errors such as insertions, deletions, and substitutions, in turn establishing the need for a model robust to the conversational properties. In this work, we adopt an approach of instruction fine-tuning (Wei et al., 2022; Longpre et al., 2023), which is fine-tuning the language model on a mixture of tasks expressed via natural language instructions.

The process of fine-tuning a LM for contact-center applications involves three main components: a contact-center dataset, instructions specific to contact center use-cases, and a language model. To curate the contact-center dataset, we collect ASR transcripts of English conversations between agents and customers from various sectors, such as e-commerce, ed-tech, logistics, etc. We observe an average word-error-rate (WER) of 14.3 on these transcripts. The next step is to gather the instructions and their corresponding responses from the collected calls. We employ three processes to obtain these instructions:

- Initially, we utilize our previously annotated data from use-cases such as sentiment detection, intent classification, entity recognition, and question answering. We reformat this data into triplets containing an *{instruction, input, output}*. The instructions and outputs for these tasks are aggregated through a semi-automatic process involving human intervention. We leverage the human-in-the-loop approach to generate instructions and corresponding responses for the given task.
- Following this, we expand the instructions by employing a paraphrasing process. This allows us to generate multiple styles of the same instructions, thereby increasing the diversity of the instruction set.
- In addition to using the annotated data from the past, we also gather new sets of instructions by instructing human annotators to generate relevant questions that can be asked and answered during a call. Similar to the previous step, we expand these generated instructions using the paraphrasing process.

To assist the annotators in generating these tasks, we provide them with a list of insights that we aim to extract from the calls to address various use-cases. Examples of such insights include understanding and tracking customer and agent behaviors, following the steps taken in the call to resolve customer issues, and identifying different objections raised by the customers. The overall corpus is constructed with a diversity of full call transcripts, segmented call transcripts and individual speaker turns. On an average each task-instruction is paraphrased into 50 alternate instruction to make the model generalizable to unseen variations.

Here are some important statistics on the internally curated contact-center dataset:

- Total corpus size: 110030
- Number of tasks: 59
- Number of instructions: 2468

Some example tasks considered in the dataset include *reason for call*, *call summarization*, *segmented call summarization*, *confirmed next steps*,

[2]We cannot release the dataset due to proprietary reasons.

Task	Task Instruction
Call Reason	What is the primary call intent
Call Summarization	Summarize the dialog
Segmented Call Summarization	Summarize a segmented portion of the dialog
Confirmed Next Steps	List the confirmed next steps if any in the dialog
Question-Answering (QA)	Answer the question based on context present in the dialog
Entity Extraction	List the entities present in the dialog
Topic Segmentation	Segment the dialog into coherent topics
Text Rewriting (QA)	Rewrite a given piece of text in a fluent and grammatically correct form
Sentiment Classification	Classify the sentiment of the customer in the call among positive, negative and neutral.

Table 3: Definitions of representative tasks considered in the internally curated contact-center dataset. These tasks were utilized as the downstream tasks for RQ1 (Section 5.1).

Question-Answering (QA), *entity extraction*, *topic segmentation*, *text rewriting*, *sentiment classification*. Refer to Table 3 for instructions used for these tasks.

Further, we fine-tune OOB-LLMs that are free for commercial use on the curated dataset. Specifically, we obtain CC-Flan-T5 model by fine-tuning the corresponding sized OOB-Flan-T5 model, and obtain CC-Llama model by fine-tuning the corresponding sized OOB-Llama-Instruct model. The models were trained on 8×A100 40GB GPUs (p4d.24x larger) using Deepspeed [3] library. The models were fine-tuned for a total of 2 epochs. The training time per epoch for Flan-T5 (11B) model is 32 hours, while it take 17 hours to train Llama (13B) for each epoch.

A.2 Implementation Details for Probing Setup (RQ2, RQ3, RQ4)

In this section, we provide a detailed account of the implementation specifics related to our investigation into LLMs fine-tuned on CC instructions.

- **Representation Extraction**: To initiate the process, we extract representations from the LLMs, harnessing their hidden states to encapsulate the contextual nuances present in the transcripts as well as instructions which are indicative of the tasks they are expected to perform as demonstrated in a previous study (Amini and Ciaramita, 2023). Our approach is different from the authors in the sense that we use a linear probe as opposed to an attentional probe which is explained in more detail later in this section. For encoder-decoder models, we tap into the final encoder layer to obtain representations for each token within the input prompt. We adopt a suitable aggregation method depending on the characteristics of the specific probing task. For single-token probing tasks, we use the representation of the target token. For other tasks, we obtain an average of representations of all input tokens. On the other hand, in decoder-only models, we utilize the last hidden layer of the decoder block. The aggregation approach for decoder-only models aligns with encoder-decoder models for single-token probing tasks but relies on the last token's representation for other tasks. This difference stems from encoder-decoder models being bidirectional, making each token representation contextual to the entire sequence. In contrast, decoder models process tokens sequentially from left to right, making each token's representation contextual only to the tokens before it. Therefore, we consider the last token's representation as it encompasses information from entire sequence.

  For encoder-decoder models, the embedding dimension spans 512, 1024, 2048, and 4096 tokens, while for decoder-only models, it encompasses 32001 and 65024 tokens. The different embedding dimensions for the two classes of models stems from the difference in model architectures and context lengths employed during pre-training and fine-tuning. We employed a context length of 512 for all models when extracting representations due to the input prompts having a maximum sequence length of 507 tokens across probing tasks. All models receive an input consisting of a prompt, which is a combination of transcript generated from the input dialog, and an instruction that defines the probing task being conducted.

- **Hyperparameters**: Post representation extraction, we employ a Multilayer Perceptron

[3] https://github.com/microsoft/DeepSpeed

(MLP) comprising a single hidden layer, utilizing the extracted representations as feature inputs for probing. We adopt a *sigmoid* and *softmax* activation function for binary and multi-class classification respectively. We perform a hyper-parameter sweep over the range - number of neurons in the hidden layer $\in \{50, 100, 150, 200\}$, learning rate $\in \{1e-3, 1e-2, 5e-2\}$, batch size $\in \{4, 8, 16, 32, 64\}$ and choose the best setting as evaluated on eval set. Additionally, we employ Adam optimizer with a dropout rate of 0.3, incorporate a weight decay of 0.00001, and set the maximum number of epochs to 20. Moreover, all experiments include early stopping and check-pointing for the best model.

- **Compute Infra**: Our experiments comprising representation extraction and probe classifier training were conducted on an AWS cloud instance, specifically, the p4d.24xlarge instance, equipped with eight GPUs, each boasting 40 GB of memory. The process of extracting representations is computationally intensive, chiefly because of the substantial embedding dimensionality. On average, a single run of the representation extraction job for decoder-only models of size 13 billion parameters demands 8-10 hours for completion, whereas the corresponding timeframe for encoder-decoder models of size 11 billion parameters is considerably shorter, ranging from 1-2 hours. In contrast, training of probing classifiers present a lighter computational load and general taking around 0.5 hours for each classifier.

- **Sample instructions used for contact-center probing tasks**
  - Disfluency Detection: Is the given spoken utterance disfluent?
  - Pause Classification: Does the speaker take long pauses while speaking?
  - Overtalk Detection: Are two speakers talking over each other?
  - Question Classification: Did the speaker ask any question?
  - Speaker Role: Who among the agent or customer is the speaker for a given utterance?
  - Response Length: Is the expected response to current utterance is short or long?
  - Turn Taking: Has speaker completed its turn?
  - Token Multi: What is the error category of word *{ref_word}* among insertion error, substitution error or no error?
  - Token Binary: Is the word *{ref_word}* correct word in the given input

  As mentioned in the previous section, these instructions are concatenated with the input (dialog or turn transcript) to obtain the representations for training the probing classifiers.

A.3 Annotation process for evaluating model responses on contact center specific downstream tasks in RQ1

In the execution of this study, an annotation protocol was established, aimed at quantifying the quality of the response on the parameters of consistency, relevance, and fluency of responses generated by the large language models. Annotation guidelines were crafted, incorporating examples to illustrate the application of quality metrics, ensuring uniformity in annotator interpretation and application of these criteria.

To prepare for this task, 7 in-house annotators were subjected to a two-week training, designed to familiarize them with the nuances of instruction following large language models and interpretation of the response quality against the input of a call transcript and an instruction. This training utilized a dataset distinct from the evaluation corpus to prevent overlap and bias. Throughout the annotation process, the origins of the model outputs were anonymized to preclude annotator bias towards any specific model. Annotation agreement was monitored and evaluated through a cross-annotator review mechanism, yielding a Fleiss' Kappa score of 0.59. This score signifies moderate inter-annotator agreement, validating the reliability of the annotation process post-training.

Upon completion of the training week, the evaluation corpus was allocated among the annotators, where each annotator had to go through all data points across all models. The final response quality was judged on the basis of majority vote of the labels provided by the annotators.

Can Abstract Meaning Representation Facilitate Fair Legal Judgement Predictions?

Supriti Vijay[*,1] and **Daniel Hershcovich**[2],
[1]Adobe, India
[2]Department of Computer Science, University of Copenhagen
supriti.vijay@gmail.com, dh@di.ku.dk

Abstract

Legal judgment prediction encompasses the automated prediction of case outcomes by leveraging historical facts and opinions. While this approach holds the potential to enhance the efficiency of the legal system, it also raises critical concerns regarding the perpetuation of biases. Abstract Meaning Representation has shown promise as an intermediate text representation in various downstream NLP tasks due to its ability to capture semantically meaningful information in a graph-like structure. In this paper, we employ this ability of AMR in the legal judgement prediction task and assess to what extent it encodes biases, or conversely, abstracts away from them. Our study reveals that while AMR-based models exhibit worse overall performance than transformer-based models, they are less biased for attributes like age and defendant state compared to gender. By shedding light on these findings, this paper contributes to a more nuanced understanding of AMR's potential benefits and limitations in legal NLP.

1 Introduction

Transformer-based language models such as BERT, T5, and GPT have ushered in a new era in NLP. These language models have demonstrated exceptional proficiency in comprehending text with their non-trivial degree of knowledge in every field, propelling them to the forefront of various language-related domains (Chalkidis, 2023). However, despite their impressive performance, language models still face challenges in dealing with context-dependent language, biases in data, and a lack of interpretability (Thakkar and Jagdishbhai, 2023). Such limitations make them unsuitable for domains like legal NLP, which have an abundance of complicated, lengthy, and contextual legal documents. Thus, a system that can capture the intricate semantics of these documents is needed. Semantic representation frameworks have proven to be a promising solution, as they allow for a more nuanced understanding of language and can capture the complex relationships between legal concepts (Abend and Rappoport, 2017; Žabokrtský et al., 2020). Abstract Meaning Representation (Banarescu et al., 2013), one such framework, represents sentence-level meaning in a directed graph-based structure, with nodes representing concepts and edges representing relationships between them. This allows for a more accurate and comprehensive analysis of legal language, which is crucial in fields such as criminal and contract law, where the slightest of ambiguities can have significant consequences. However, limited knowledge exists about how useful these representations are in legal judgement prediction and whether they capture cultural and societal biases along with significant information.

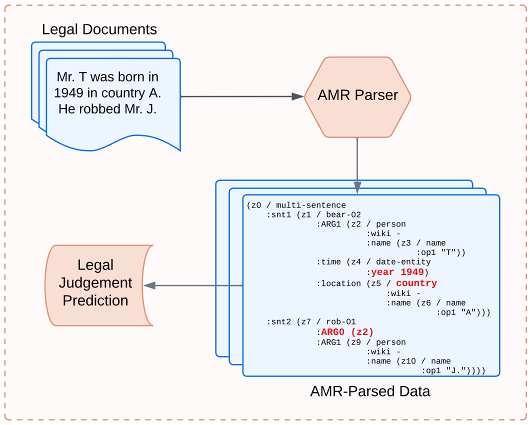


Figure 1: Abstract Meaning Representation in legal judgement prediction (LJP). Here, we demonstrate how AMR parses sensitive attributes like *age, gender identity and defendant state* as well as its ability to resolve co-referential pronouns like *he*, abstracting away gender.

This paper conducts a theoretical analysis of Abstract Meaning Representation, scrutinizing its potential in the realm of law. More concretely, it investigates the critical question of whether AMR can help produce fair legal decisions and reports

[*]Work done while at Manipal Institute of Technology

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 101–109
June 20, 2024

potential biases that may arise from its use. This evaluation helps us determine if AMR is a suitable intermediate representation for legal judgement prediction. We conduct our experiments on the ECtHR Dataset (see Section 4.1), a benchmark for legal judgement prediction which has been annotated with demographic and diversity labels. It proves to be a primary choice due to its inclusion of these labels for fairness evaluation. We utilize the macro F1 score as an evaluation metric for our experiments.

Contributions. We compare AMR's performance parity across different attributes of the ECtHR dataset, including age, gender identity, and defendant state. Our findings reflect that AMR is unable to produce fair outcomes and acts as a random baseline here. While it does report less group disparity for demographic attributes like age and state, it exhibits a low overall performance and a lower worst-case performance. We also release AMR-based models (LegalBERT and DistilRoBERTa) to enable further exploration of AMR in the legal domain.[1]

2 Related Work

While previous research has predominantly focused on AMR parsing of legal documents (Trong and Le, 2018; Vu et al., 2022; Dias et al., 2022), limited attention has been given to assessing AMR's performance in legal tasks. A study by Schrack et al. (2022) explores AMR's ability to identify logical relationships in legal MCQA tasks, revealing challenges posed by AMR parsing. In contrast, our work is the first to investigate whether AMR representations capture social biases alongside linguistic information, emphasizing the need to scrutinize AMR input representations for potential biases in legal judgement prediction tasks.

Research on fairness in machine learning models within the legal domain has also been limited. Previous studies (Angwin et al., 2016; Rice et al., 2019; Wang et al., 2021; Baker Gillis, 2021; Gumusel et al., 2022; Matthews et al., 2022; Wu et al., 2020) have highlighted racial and gender biases in the legal system and language models. More recently, Chalkidis et al. (2022) introduced the FairLex benchmark to assess the fairness of language models. In our study, we leverage one of these datasets to examine whether AMR-based models can effectively mitigate bias, addressing the critical issue of bias reduction in legal language processing.

[1] https://github.com/SupritiVijay/AMR-for-Legal-AI.

3 Abstract Meaning Representation

Abstract Meaning Representation is a structured framework that utilizes graph-like structures to represent sentence meaning, ensuring interpretability for machines and humans. These graphs, conforming to rooted, directed, and acyclic properties, are independent of semantics, grounded in syntax, and annotated using PENMAN notation for textual representation.

For example, the sentence *"Mr. T was born in 1949 in country A. He robbed Mr. J."*, as shown in Figure 1. Here, the sentence can be seen divided into two sub-sentences *(snt1 and snt2)*. In snt1, the event of "being born" *(bear-02)* is associated with Mr. T along with the : *time* and : *location* of birth. While in snt2, the event of "robbing" *(rob-01)* is described. Here, AMR can be seen establishing relationships between entities, connecting Mr. T to both the birth and the act of robbery.

4 Experimental Setup

4.1 Dataset and Metrics

The European Court of Human Rights (ECtHR) dataset (Chalkidis et al., 2021) is a text classification dataset annotated with multiple labels, which map human rights articles potentially violated in each case. It contains 11k legal cases and judgements, which are split into training (9k, 2001–16), development (1k, 2016–17), and test (1k, 2017–19) sets. Additionally, it includes distinct group tags like age, gender and defendant state for each case (See distribution in Appendix A.1). Due to its large sample size, diverse legal texts, and broad attribute coverage, ECtHR is ideal for assessing bias in AMR-based legal judgment prediction.

For a fair comparison with prior work, we adopt the same metrics used by Chalkidis et al. (2022). These include the average macro-F1 score ($mF1$), the group disparity (GD), and the worst-group performance ($mF1_W$). The $mF1$ represents the average macro-F1 score across different groups, providing a comprehensive measure of algorithm performance. The GD is calculated as the group-wise standard deviation, indicating the extent of disparity among the groups. Additionally, the worst-group performance ($mF1_W$), represents the lowest macro-F1 score among the individual groups. This allows us to gauge how poorly the most biased groups may perform.

4.2 AMR Parsing

AMR parsing has been considered a significant bottleneck (Schrack et al., 2022), especially concerning the loss of information in long and multisentence paragraphs. Hence, to overcome this challenge, we utilize the following two pre-processing techniques for our experiments.

1. Splitting before parsing (SbP): This approach involves splitting each case in the dataset into sentences before parsing, resulting in single-sentence graphs, as shown in Figure 3. These graphs are then combined to form a multi-sentence graph for a case. While this approach offers advantages in scalability, it may have limitations in terms of maintaining coherence across paragraphs.

2. Splitting after parsing (SaP): In contrast, this alternative approach focuses on creating multi-sentence graphs first, which are then linearized and split into 512 token segments to be sent to the encoder. These graphs capture interdependencies and connections between sentences, enhancing their richness compared to pooling single-sentence graphs. However, it may require more computational resources and time, as illustrated in Table 3.

We utilise the `SpringAMR` parser (Blloshmi et al., 2021) for parsing documents due to its strong and robust parsing quality. It employs a simple Seq2Seq architecture employing a pretrained BART model, trained on the Text-to-AMR task. We further explore the above techniques quantitatively and qualitatively in Appendix C.1 & C.2.

4.3 Baselines

To classify AMR-parsed graphs, we adopt a hierarchical BERT-based architecture similar to Chalkidis et al. (2022), which has been established as the benchmark model for fairness evaluation in legal datasets. This architecture effectively captures the contextual dependencies in legal documents by giving utmost attention to both paragraph and document-level representations. A detailed explanation of fine-tuning the models can be found in Appendix B. Further, we also reproduce the results of the hierarchical architecture with text-only input to evaluate the performance of AMR-based techniques in the subsequent experiments.

4.4 AMR-based models

We utilize `legalbert-base-uncased` and `distilroberta-base`, classifiers trained on textual data, as the primary models in the hierarchical architecture. We also execute continued pre-training on AMR graphs to enhance the performance of transformer models, specifically LegalBERT. We name this model as Dataset-specific LegalBERT$_{SMALL}$. Through this, we examine whether pre-training on AMR graphs captures intricate structural and semantic intricacies inherent to legal language and performs better than other classifiers. We utilize the LegalBERT model as the backbone for pretraining. This model is pre-trained using the ECtHR training subset, employing a sequence length of 128 sub-words for 10 epochs. The AdamW optimizer is used with a maximum learning rate of $1e-4$ and a 10% warm-up ratio.

5 Result Analysis

5.1 Dataset-specific vs Basic Models

In this subsection, we compare the performance of dataset-specific LegalBERT and basic LegalBERT within AMR SaP. The mF1-scores in Table 1 show a significant performance decline with pre-training, attributed to introduced noise and biases inherent in the dataset. In contrast, the basic LegalBERT model, which is trained directly on the specific legal classification task without the additional step of pre-training, can solely focus on learning from the task-specific data. Additionally, we observe that a generalized adaptation to legal knowledge may be more effective than attuning a pre-trained model on the experimental dataset. The vast overview of legal knowledge assists the basic model in acquiring a strong foundation in legal language understanding, allowing it to outperform the dataset-specific model.

5.2 Fairness Analysis

Analysing the results presented in Table 1, it becomes evident that the benchmark DistilRoBERTa$_{FairLex}$ model displays notable group disparities, particularly for Defendant State and Applicant Age. In contrast, most AMR-based models exhibit reduced group disparities in these attributes. However, when it comes to Applicant Gender, AMR-based models consistently demonstrate higher group disparities, with LegalBERT$_{SMALL}$ (AMR SbP) recording the highest GD for it. This phenomenon may be attributed to the parsing of individual sentences, assigning equal weight to all words, including gendered ones, potentially perpetuating implicit biases within the model. In the broader context, we identify a recurring trend where AMR-based

Language Models	Average mF1	*Defendent State* mF1 ↑	GD ↓	mF1_w ↑	*Applicant Gender* mF1 ↑	GD ↓	mF1_w ↑	*Applicant Age* mF1 ↑	GD ↓	mF1_w ↑
ECtHR (ECHR Violation Prediction)										
Text Based Models										
DistilRoBERTa	62.9	63.3	2.1	61.2	59.0	2.0	56.3	61.3	2.5	58.5
$\text{DistilRoBERTa}_{FairLex}$	NA	53.2	8.3	44.9	57.5	3.1	54.4	54.1	5.9	46.2
AMR Split before Parsing										
LegalBERT_{SMALL}	54.8	50.5	1.2	49.3	47.1	5.4	40.4	52.4	4.8	47.2
AMR Split after Parsing										
LegalBERT_{SMALL}	57.3	**59.2**	**0.3**	**58.8**	**56.0**	3.5	**52.3**	**56.5**	3.7	**50.1**
(Dataset-specific LegalBERT_{SMALL})	44.2	40.4	5.3	35.0	32.1	**2.5**	28.9	33.3	**0.8**	31.9
DistilRoBERTa	37.6	36.5	0.7	35.7	31.6	4.4	28.3	36.2	5.4	27.6

Table 1: Test results for different baselines and models per ECtHR attribute. We report the average performance across groups (mF1), the group disparity (GD), and the worst-group performance (mF1_w). ↑ denotes that higher scores are better and ↓ vice versa. We report results by Chalkidis et al. (2022) as $\text{DistilRoBERTa}_{FairLex}$.

models exhibit higher fairness levels compared to text-based models. However, this advantage is offset by lower $mF1$ scores and overall performance metrics. Notably, a subset of AMR-based models, primarily LegalBERT_{SMALL} (AMR SbP), approaches the performance of text-based models but lacks consistency in addressing group disparities across all attributes.

Digging deeper into worst-case performance, we notice that while AMR models inherently prioritize fairness, their lower worst-case performance scores render them impractical for real-world applications. This raises a crucial question: *does a model with greater fairness, at the cost of overall performance, hold value?*. In essence, a model with zero performance yields zero group disparity. This brings to light a paradox: the fairness demonstrated by AMR models, despite having low group disparity, takes on the semblance of a random baseline due to its lack of substantial performance metrics. Consequently, we assert that AMR may not be the optimal choice for ensuring fairness in practice.

5.2.1 Potential Biases

As illustrated in Table 1, we observe that AMR-based models demonstrate lower group disparity than the benchmark $\text{DistilRoBERTa}_{FairLex}$ model for defendant state and applicant age and higher group disparity for Applicant Gender. This could be attributed to the fact that other group identifiers, such as defendant state and age, may not be directly linked to the individual during AMR parsing.

For example, the sentence "Mr. T was born in 1949 in country A. He robbed Mr. J." as represented in Figure 1. Here, the accurate recording of the applicant's country (location-$z5$) and year (time-$z4$) establishes a direct link with *:ARG1*-$z1$, while coreference in $z7$ is directly associated with *:ARG0*-$z2$. This distinction implies that while coreferences consistently refer to the individual, contextual details such as time and location are connected to the event itself. Consequently, the presence of pronouns in the case establishes a direct relationship between the gender and personal information of the individual. This dissociation between these contextual elements and the individual prevents the subsequent classification model from making inferences based on these attributes. As a result, age and defendant state exhibit lower group disparity, while gender disparity remains consistent throughout the analysis.

6 Conclusion

In this paper, we explore the application of Abstract Meaning Representation (AMR) in predicting legal judgments. Our analysis has revealed both the benefits and challenges associated with using AMR in this context. While AMRs offer the capability to capture the semantics of legal texts and enable automated analysis and decision-making, providing a promising avenue for fair judgement still remains ambiguous in domains like Applicant Gender. Even so, it clearly demonstrates its efficacy in other group disparities like Age and Defendant State. However, due to their poor performance and low mF1 scores, we conclude that while AMR-based models are fairer by design, they are unsuitable for ensuring fairness in the real world.

Limitations

We experimented with one AMR parser (with two sentence-splitting strategies), `SpringAMR`. While this is a widely used and highly accurate AMR parser, other parsers might exhibit different behavior with respect to encoding demographic attributes such as those we investigate here.

Furthermore, while AMR is the most popular meaning representation framework, other meaning representation frameworks may again behave differently. For example, UCCA (Abend and Rappoport, 2013) represents semantic structure without attempting to capture lexical disambiguation at all.

Finally, we only investigated one of the datasets included in FairLex, namely ECtHR, targeting the age, defendant state and gender attributes. Different conclusions may be drawn regarding other datasets, tasks and attributes—for example, the SCOTUS dataset indicates whether the respondent is a person, public entity, organization, facility or other. FSCS contains the language and region of the case. Further investigation is required to better understand and address the limitations of what is represented in the parsed AMRs and what is not to ensure fair and accurate predictions across all demographic groups.

Ethics Statement

Automating legal judgement prediction raises ethical implications and warrants a thorough examination of potential biases. Our AMR-based models have shown promising improvements in group disparity. However, the parsed AMR may nevertheless unintentionally overlook or misrepresent certain group identifiers, leading to biased predictions we are not yet aware of. Furthermore, the remaining performance disparities observed across demographic groups, particularly in Applicant Gender, highlight the need for continuous evaluation, improvement in fairness considerations and stronger guarantees before deploying such models in legal contexts.

The ECtHR dataset is released as part of FairLex under the CC-BY-NC-SA-4.0 license. We only use it for our experiments and do not redistribute it. Furthermore, the original dataset is anonymized, and we do not add any new data—particularly no personal information.

https://creativecommons.org/licenses/by-nc-sa/4.0/

Acknowledgements

References

Omri Abend and Ari Rappoport. 2013. Universal Conceptual Cognitive Annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 228–238, Sofia, Bulgaria. Association for Computational Linguistics.

Omri Abend and Ari Rappoport. 2017. The state of the art in semantic representation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 77–89, Vancouver, Canada. Association for Computational Linguistics.

Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. 2016. Machine bias: There's software used across the country to predict future criminals. *And it's biased against blacks. ProPublica*, 23:77–91.

Noa Baker Gillis. 2021. Sexism in the judiciary: The importance of bias definition in NLP and in our courts. In *Proceedings of the 3rd Workshop on Gender Bias in Natural Language Processing*, pages 45–54, Online. Association for Computational Linguistics.

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.

Rexhina Blloshmi, Michele Bevilacqua, Edoardo Fabiano, Valentina Caruso, and Roberto Navigli. 2021. SPRING Goes Online: End-to-End AMR Parsing and Generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 134–142, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ilias Chalkidis. 2023. ChatGPT may pass the bar exam soon, but has a long way to go for the LexGLUE benchmark.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, Dimitrios Tsarapatsanis, Nikolaos Aletras, Ion Androutsopoulos, and Prodromos Malakasiotis. 2021. Paragraph-level rationale extraction through regularization: A case study on european court of human rights cases.

Ilias Chalkidis, Tommaso Pasini, Sheng Zhang, Letizia Tomada, Sebastian Schwemer, and Anders Søgaard.

2022. FairLex: A multilingual benchmark for evaluating fairness in legal text processing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4389–4406, Dublin, Ireland. Association for Computational Linguistics.

Martha Chamallas. 2019. Feminist legal theory and tort law. In *Research Handbook on Feminist Jurisprudence*, pages 386–405. Edward Elgar Publishing.

João Dias, Pedro A. Santos, Nuno Cordeiro, Ana Antunes, Bruno Martins, Jorge Baptista, and Carlos Gonçalves. 2022. State of the art in artificial intelligence applied to the legal domain.

Ece Gumusel, Vincent Quirante Malic, Devan Ray Donaldson, Kevin Ashley, and Xiaozhong Liu. 2022. An annotation schema for the detection of social bias in legal text corpora. In *Information for a Better World: Shaping the Global Future: 17th International Conference, iConference 2022, Virtual Event, February 28–March 4, 2022, Proceedings, Part I*, pages 185–194. Springer.

Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. 2018. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1929–1938. PMLR.

Sean Matthews, John Hudzina, and Dawn Sepehr. 2022. Gender and racial stereotype detection in legal opinion word embeddings.

Douglas Rice, Jesse H. Rhodes, and Tatishe Nteta. 2019. Racial bias in legal language. *Research & Politics*, 6(2):2053168019848930.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Jackson Sargent and Melanie Weber. 2021. Identifying biases in legal data: An algorithmic fairness perspective.

Nikolaus Schrack, Ruixiang Cui, Hugo López, and Daniel Hershcovich. 2022. Can AMR assist legal and logical reasoning? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1555–1568, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Krishna Thakkar and Nimit Jagdishbhai. 2023. Exploring the capabilities and limitations of gpt and chat gpt in natural language processing. *Journal of Management Research and Analysis*, 10:18–20.

Sinh Vu Trong and Minh Nguyen Le. 2018. An empirical evaluation of AMR parsing for legal documents.

Sinh Trong Vu, Minh Le Nguyen, and Ken Satoh. 2022. Abstract meaning representation for legal documents: an empirical research on a human-annotated dataset. *Artificial Intelligence and Law*, pages 1–23.

Yuzhong Wang, Chaojun Xiao, Shirong Ma, Haoxi Zhong, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2021. Equality before the law: Legal judgment consistency analysis for fairness. *arXiv preprint arXiv:2103.13868*.

Yiquan Wu, Kun Kuang, Yating Zhang, Xiaozhong Liu, Changlong Sun, Jun Xiao, Yueting Zhuang, Luo Si, and Fei Wu. 2020. De-biased court's view generation with causality. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 763–780, Online. Association for Computational Linguistics.

Zdeněk Žabokrtský, Daniel Zeman, and Magda Ševčíková. 2020. Sentence meaning representations across languages: What can we learn from existing frameworks? *Computational Linguistics*, 46(3):605–665.

A Fairness in Legal Judgement Prediction

The legal domain represents a complex and multifaceted system shaped by various social, cultural, and historical factors. Portrayed as blind, unbiased, and objective, justice is often plagued by systemic biases ingrained in the language of judicial opinions, case outcomes, and the personal predispositions of its practitioners (Rice et al., 2019). While for NLP in law, these biases manifest in either representational harms where certain social groups are over or underrepresented or sentencing disparities across certain groups (Sargent and Weber, 2021). In our evaluation of fairness, we adopt an *equal risk or equal odds* (Hashimoto et al., 2018) approach where we define bias as the disproportionate performance of a classifier across different groups with similar risk profiles. Such parity conclusively establishes sensitive traits like *age, nationality, and gender* as significant attributes when forming an outcome. Therefore, we embrace this asymmetry in efficacy as a measure of fairness across input representations in the legal judgement prediction domain.

For instance, victims of domestic violence, rape, and sexual assault have little recourse to obtain tort compensation due to the installation of recovery restrictions (Baker Gillis, 2021; Chamallas, 2019). This is merely one situation where failing to provide equal weight to all genders in the law results in severe damage.

A.1 FairLex & the ECtHR dataset

We use prior work conducted under FairLex (Chalkidis et al., 2022) as our baseline for text

Applicant Age				Applicant Gender			Defendant State	
N/A	>35	<65	>65	N/A	Male	Female	E.C.	West
2,794	839	4,246	1,121	3,306	4,407	1,287	7,224	1,776

Table 2: Group distribution in training set for each attribute of ECtHR dataset. These are the statistics presented in the FairLex paper (Chalkidis et al., 2022).

classification and fairness. The study partitions the ECtHR dataset on the following attributes:

1. **Defendant States:** These comprise European nations accused of breaching the ECHR. Each case's defendant states form a subset of the 47 Council of Europe Member States. To establish statistical significance, the defendant states are categorized into two groups: Central-Eastern European states and all other states, as delineated by the EuroVoc thesaurus.

2. **Applicant's Age:** The applicant's birth year is gleaned from case facts whenever possible, leading to classification within age groups ($\leq$ 35, $\leq$ 64, or older).

3. **Applicant's Gender:** Extracted from case details, gender is categorized as male or female based on pronouns or other gender-specific terminology. We will add these attribute distributions to the dataset description as well.

B Problem Formulation

In this section, we introduce the notations used for the task of predicting legal judgments. Let $(X_i, Y_i)_{i=1}^{N}$ represent a training set comprising N samples. Each sample consists of an input list of facts denoted as $X_i = \{t_1, t_2, ..., t_m\}$, pertaining to a single legal case. To capture the semantic and relational nature of the text, we feed these text paragraphs into an AMR parser, which generates the respective graphs, i.e., each t_j creates its own encoded graph f_j. Therefore, if initially each sample was represented by $X_i = \{t_1, t_2, ..., t_m\}$, where each X_i was an entire legal case and each t_j were its individual facts, after encoding by AMRs, they can be represented as $X_i = \{f_1, f_2, ..., f_m\}$. With this, we have restructured the problem statement as judgement prediction using AMR-graphs. The corresponding labels for the multi-label classification task are represented by $Y_i = \{y_1, y_2, ..., y_{10}\}$. Our objective is to maximize the posterior probability $p(Y|X)$ for each case. However, due to the presence of lengthy textual content within each case and the inherent token limit of transformer-based language

	Parsing Time (seconds)	Average No. of Tokens (case)
Split Before	444960	47387.96
Split After	648000	68439.15

Table 3: Statistics for the two parsing strategies: sentence splitting before/after parsing.

models, we adopt a hierarchical approach to address this challenge.

This architecture uses a transformer-based backbone model, such as LegalBERT (Chalkidis et al., 2020) or DistilRoBERTa (Sanh et al., 2020), to generate embeddings for each fact (f_k) in the input. This enables us to obtain contextualized representations for each fact. Instead of using pooling techniques at the word level, we consider the representation of the $[CLS]$ token as the fact embedding (e_k), capturing the global context of the entire fact. Subsequently, a segmentation-encoder layer is employed to process the fact embeddings ($E = \{e_1, ..., e_k, ..., e_m\}$) and capture the longform structure of the legal case. This layer combines the fact embeddings using attention weights, generating a multi-vector representation for each fact in the case ($SE = \{se_1, ..., se_k, ..., se_m\}$). These representations are then pooled and fed into a classification layer to generate the probability (p) of a violation (Y) given the input (X).

C AMR Parsing

C.1 Quantitative Analysis

We compare the length of parsed strings using two AMR parsing techniques, "Splitting before parsing" (X-axis) and "Splitting after parsing" (Y-axis), as shown in Figure 2. The plot illustrates a significant difference, with a distinct upper bound on the Y-axis (1.4M characters) and a lower bound on the X-axis (391k characters), taking into account characters and whitespaces. This trend persists even after removing whitespaces using regex, indicating that "Split After" consistently results in longer strings. Additionally, we compute statistics on depth and the

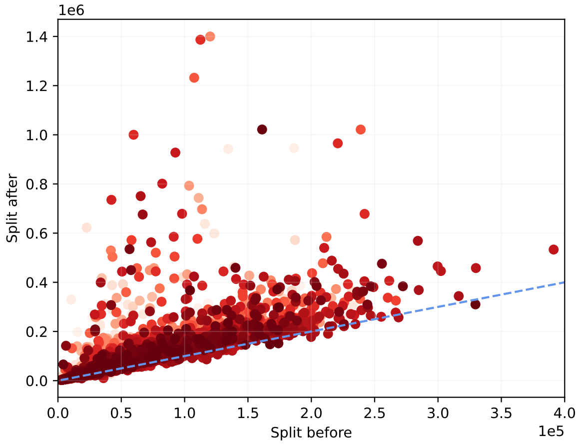


Figure 2: A scatter plot depicting the length of parsed strings using AMR parsing techniques, "Splitting before parsing" (X-axis) and "Splitting after parsing" (Y-axis), reveals a noticeable difference between them. The plot shows a wider dispersion of data points on the Y-axis. Here, length of parsed strings refer to the number of characters in the entire case, i.e., all linearized graphs that are concatenated together.

number of relations. The average depth for "Split Before" is 17.76, while for "Split After," it is 44.81, suggesting higher structural complexity in the latter. Likewise, "Split After" exhibits an average number of relations at 44.90, compared to 17.87 for "Split Before," indicating more interactions in the former.

Additionally, the scatter plot demonstrates that the “Splitting after parsing” technique exhibits a wider dispersion of data points on the Y-axis, indicating its ability to retain a more significant amount of knowledge. These findings highlight the effectiveness of the “Splitting after parsing” technique in capturing more information.

C.2 Qualitative Analysis

In this section, we study different techniques of parsing from the perspective of structure, coreference, and context retention. The first technique, “Splitting before parsing,” offers scalability, although it also limits context understanding and coherence across paragraphs. For instance, as shown in Figure 3, individual sentences may not capture the associations between entities, leading to a lack of comprehensive analysis. Furthermore, we observe that while splitting a paragraph into component sentences, certain short phrases enclosed between two periods tend to be skipped. In the example presented in Figure 3, person "T." can be seen eliminated during graph generation. We have validated these errors in “Splitting before parsing” method using a naive approach of '.' detection, as well as, using NLTK's $sentence - splitter$. The issue of co-reference still persists across both splitters as expected.

In contrast, the second technique, “Splitting after parsing,” retains entity and event-coreference and maintains a stronger connection to the original context. Here, splitting is based on the limitation provided by the LLM model, since we are using $BERT$, it is the $max - tokens$ which can be fed into that model. This, allows the graphs to strongly associate and encode large amounts of text data, including their co-references irrespective of the sentence structure. Upon feeding it further for classification, since we use a HAN architecture it continues to carry-forward the same co-references in its predictions. Therefore, as demonstrated in Figure 3, the multi-sentence graph represents the same content but with a different organization, capturing diverse associations and temporal relationships. It is able to better capture the interrelation between the individuals involved, the event, and the timing of the event. This technique contributes to more accurate parsing results and a deeper understanding of legal entities and their relations.

While our findings suggest the "Splitting after parsing” method is a more effective parsing strategy for AMR graphs, we still witness occasional oversights by the approach. Such as the graph on the left (*split before*) uses the same variable $z0$ for the person "J.", the action of placing, and the action of visiting. This is incorrect as they are distinct entities or events. The person "T." who visited is not represented in the graph. The graph does not capture that both the placing and the visiting happened on the same day, 23 June 1993. The graph uses ($z1$ / she) to represent "her," but it's not clear that "her" refers to "J". The graph separates the events of placing and visiting into different sub-graphs but does not establish any relationship between them. Also, the date "23 June 1993" is associated only with the person "J." and not with the events of placing and visiting. The graph on the right (*split after*) uses a single variable $z1$ to represent both "J." and "T." under $: name$. This is incorrect as they are distinct entities. While the graph includes the date entity $z6$, it is only linked to the $place - 01$ event. It should also be linked to the $visit - 01$ event to indicate that both events happened on the same day. The graph still does not make it clear that "her" refers to "J." Coreference should be explicitly represented.

31. On 23 June 1993 J. was placed in the family centre. T. visited her the same day.

Split-Before

```
(z0 / person
    :wiki -
    :name (z1 / name
              :op1 "J")
    :time (z2 / date-entity
              :day 23
              :year 1993))
(z0 / place-01
    :ARG2 (z1 / center
              :mod (z2 / family)))
(z0 / visit-01
    :ARG1 (z1 / she)
    :time (z2 / day
              :ARG1-of (z3 / same-01)))
```

Split-After

```
(z0 / visit-01
    :li 31
    :ARG0 (z1 / person
              :wiki -
              :name (z2 / name
                        :op1 "J."
                        :op2 "T."))
    :ARG1 (z3 / place-01
              :ARG1 z1
              :ARG2 (z4 / center
                        :mod (z5 / family))
              :time (z6 / date-entity
                        :year 1993
                        :month 6
                        :day 23
                        :time-of z0)))
```

Figure 3: AMR graphs, in PENMAN format, obtained through sentence splitting before (left) and after parsing (right), showing the differences in graph structure. In the former, sentence splitting errors result in an incorrect AMR. The latter results in an AMR with less severe errors, which also demonstrates cross-sentence co-reference resolution of the time expression. For distinction, we present segments of the image in red, which are clearly contrasted within the “Split-Before” side of the image. We see that "T.", "month 6", and "time-of z0" are better co-referenced and associated by the “Split-After” technique.

WinoViz: Probing Visual Properties of Objects Under Different States

Woojeong Jin, Tejas Srinivasan, Jesse Thomason, Xiang Ren
Department of Computer Science, University of Southern California, USA
{woojeong.jin,tejas.srinivasan,jessetho,xiangren}@usc.edu

Abstract

Humans interpret visual aspects of objects based on contexts. For example, a banana appears brown when rotten and green when unripe. Previous studies focused on language models' grasp of typical object properties. We introduce WinoViz, a text-only dataset with 1,380 examples of probing language models' reasoning about diverse visual properties under different contexts. Our task demands pragmatic and visual knowledge reasoning. We also present multi-hop data, a more challenging version requiring multi-step reasoning chains. Experimental findings include: a) GPT-4 excels overall but struggles with multi-hop data. b) Large models perform well in pragmatic reasoning but struggle with visual knowledge reasoning. c) Vision-language models outperform language-only models.

1 Introduction

Language models (LMs) face challenges in developing intuitive reasoning and acquiring knowledge from experience, similar to humans. Human knowledge acquisition from the visual world is effortless but poses difficulties for LMs, as such knowledge is often not explicitly described in text. Overcoming these challenges requires visual grounding, connecting language and visual information for comprehension.

Previous studies have predominantly aimed at investigating language models in relation to object prototypical visual properties such as color, shape, and affordance, and transferring such knowledge from vision-language models (Norlund et al., 2021; Paik et al., 2021; Zhang et al., 2022; Li et al., 2023b). In this work, we study language models' reasoning ability on associations between objects and their visual properties across different object states. The task requires a model to reason about different states of an object where the object may exhibit different properties.

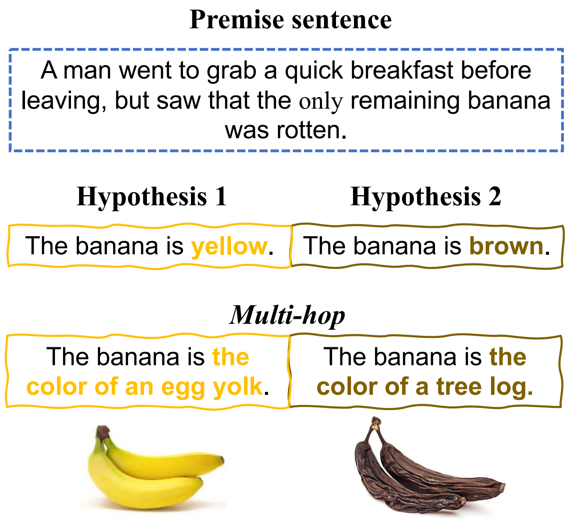


Figure 1: **The WinoViz task.** We investigate the divergent properties of an object and explore the reasoning abilities of language models pertaining to object attributes.

In this work, we investigate the divergent properties of an object and explore the reasoning abilities of language models pertaining to object attributes. Annotators create a premise sentence portraying a scene with a banana and two hypothesis sentences highlighting its visual properties as depicted in Fig. 1. The goal is to choose a more plausible hypothesis, requiring comprehension of the banana's properties in different states. A more challenging multi-hop version replaces the visual attribute word with another object word sharing a similar visual attribute.

Benchmarking zero-/few-shot performance includes text-only models like BERT (Kenton and Toutanova, 2019), T5 (Raffel et al., 2020; Chung et al., 2022), and GPT variants (Brown et al., 2020), ranging from 110 million to 175 billion parameters. Models incorporating visual information, such as VL-BERT (Su et al., 2019) and Oscar (Li et al., 2020), are explored.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 110–123
June 20, 2024 ©2024 Association for Computational Linguistics

Key findings from experiments with the WINOVIZ benchmark include: a) GPT-4 performs effectively but degrades on multi-hop data. b) Large models excel in pragmatic reasoning but face challenges in visual knowledge reasoning. c) Vision-language models outperform language models.

2 The WINOVIZ Task

The WINOVIZ task entails the need for a model to deduce whether objects can demonstrate prototypical behaviors in various scenarios. More precisely, when provided with a natural language sentence describing an object engaged in a particular behavior (*premise sentence*), the model must determine between two sentences presenting contrasting visual attributes of the object (*hypothesis sentences*). Fig. 2 includes dataset collection (details are in the appendix)

Challenges. The WINOVIZ task assesses a machine's reasoning ability about daily objects, focusing on their varied properties. Models often struggle with visual knowledge related to common objects due to limited explicit details in training text, attributed to reporting bias (Norlund et al., 2021; Jin et al., 2022). The task is challenging as it requires pragmatic reasoning and visual knowledge reasoning, involving finding intended meanings in the text and reasoning about object properties. A more challenging version, *multi-hop data*, requires multi-step reasoning chains.

3 Experiments

We first describe the experimental setup used in our analysis and share experimental results.

Language Models. We experiment with 7 language models in total (Table 5). We include encoder-only, encoder-decoder, decoder-only models. The sizes of LMs vary from 109M to 175B. We include large LMs, GPT-3, GPT-3.5, and GPT-4 (Brown et al., 2020; Ouyang et al., 2022; OpenAI, 2023).

Vision-language Models. We experiment with a total of 5 vision-language models (see Table 5). Our task involves understanding visual information about objects in various states, derived from image-caption datasets. We investigate whether vision-language models surpass language models in our task. For evaluation, we deliberately exclude

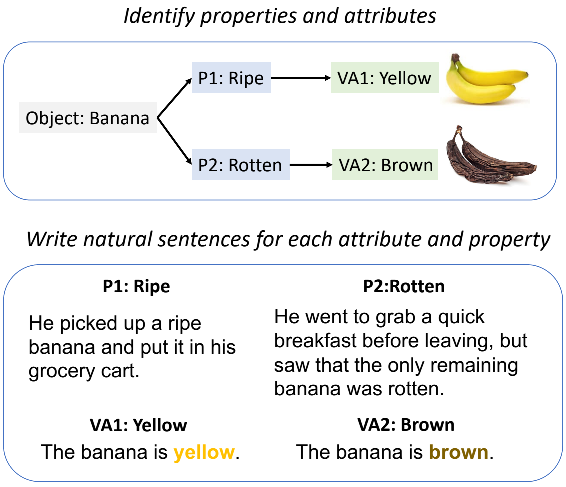


Figure 2: **Dataset Collection.** We collect our data through crowdsourcing efforts. The first step is to identify properties and visual attributes for an object and the second step is to write natural sentences for each property and attribute. Sentences with properties will be used as premise sentences and sentences with visual attributes will be used as hypothesis sentences.

image inputs and focus solely on the language components of the models, using encoder-only models (VL-BERT (Su et al., 2019) and Oscar (Li et al., 2020)), a decoder-only model (LLaVA-v1.5 (Liu et al., 2023)), and a bi-encoder model (CLIP 'clip-vit-large-patch14'(Radford et al., 2021)).

Inference. In our analysis, we rely on zero-shot inference and few-shot in-context learning for encoder-decoder, decoder-only models. Our prompt design for the zero-shot inference is as follows: *"You will be given a sentence, and two options. Output either Option 1 or Option 2, depending on which option is more likely to be true given the sentence."* For the few-shot in-context learning, we use 4 examples. We also adopt chain-of-thought prompting (Wei et al., 2022) for the few-shot inference. In addition to the encoder-decoder and decoder-only models, we explore encoder-only models. Encoder-only models cannot do zero-shot inference for multi-choice tasks since it requires a task-specific head for unseen tasks. Thus, we fine-tune the encoder-only models with SNLI (Bowman et al., 2015) and ANLI (Nie et al., 2019) datasets and we use only 'contradiction' and 'entailment' labels in fine-tuning.

Evaluation Setup. We evaluate models with two different metrics: individual accuracy (Ind.) and pair accuracy (Pair). Individual accuracy refers to accuracy on each individual question, while pair

Model	Single-hop		Multi-hop	
	Ind.	Pair	Ind.	Pair
FLAN-T5-XXL	86.24	72.71	68.09	40.43
LLaMA2	73.28	48.85	52.84	20.45
LLaVA	79.47	59.63	56.82	17.05
GPT-3	84.17	69.24	58.5	22
GPT-3.5	86.58	75.62	58	20
GPT-4	90.25	81.19	72	45

Table 1: **Results on WINOVIZ in a zero-shot manner.** We evaluate large models using 0 examples on both our single-hop and multi-hop datasets. We observe that these models performed well on the single-hop data; however, their performance is significantly degraded on the multi-hop data.

accuracy refers to the accuracy on each pair of questions. In WINOVIZ, two premise sentences are paired and they share the same set of hypothesis options. We measure the model's performance based on its ability to accurately predict both premise sentences. If the model's prediction is correct for only one of the premise sentences in the pair, we consider the prediction less robust.

3.1 Analysis Questions

In our empirical analysis, we try to answer the following questions:

1. How good are large models on our task? When it comes to multi-hop data, how good are they? (Section 3.2)

2. Do few-shot prompting and CoT prompting improve the results? (Section 3.3)

3. Which reasoning step between pragmatic reasoning and visual knowledge reasoning is main bottleneck in our task? (Section 3.5)

4. Do vision-language models outperform language-model counterparts? (Section 3.2)

3.2 Zero-shot Results

We evaluate language models and vision-language models in a zero-shot way, without utilizing any training data (Table 1). Overall, large models perform well on the single-hop data, but their performance is significantly degraded on the multi-hop data. Among them, GPT-4 exhibits the best overall performance on both single-hop and multi-hop tasks. Surprisingly, FLAN-T5-XXL, the smallest model among the comparison, yields comparable results to larger models, including GPT-3. Moreover, it outperforms GPT-3 and GPT-3.5 on the multi-hop dataset. LLaVA, built upon LLaMA2 and trained with image-caption datasets, shows noteworthy performance. As indicated in the table, LLaVA surpasses LLaMA2 on both single-hop and multi-hop data, suggesting that image-caption datasets enhance reasoning in our task.

Model	Single-hop		Multi-hop	
	Ind.	Pair	Ind.	Pair
FLAN-T5 (0)	86.35	73.17	68.09	40.43
FLAN-T5 (4)	87.84	76.15	69.32	42.05
FLAN-T5 (4 CoT)	87.16	74.77	67.05	38.64
GPT-3.5 (0)	86.58	75.62	58	20
GPT-3.5 (4)	88.42	77.75	62.5	28.41
GPT-3.5 (4 CoT)	77.18	59.63	65.34	34.09

Table 2: **Results on WINOVIZ with 4-shot in-context learning.** We use FLAN-T5-XXL and GPT-3.5 in this analysis. Standard prompting marginally improves the performance of them, while chain-of-thought prompting is beneficial for GPT-3.5 in the multi-hop task.

Method	Single-hop		Multi-hop	
	Ind.	Pair	Ind.	Pair
BERT-Large	67.31	39.44	54	16
VL-BERT-Large	69.61	42.88	56	18
Oscar-Large	72.93	50.22	64.5	32

Table 3: **Results on WINOVIZ after NLI training.** We train encoder-only models on NLI datasets and choose an option by the highest probability of the 'entailment' class.

3.3 Few-shot Results

Table 2 displays the results with 4 in-context examples for FLAN-T5-XXL and GPT-3.5. We conduct tests using standard prompting and chain-of-thought prompting in this experiment. Initially, standard prompting with 4 in-context examples marginally improves the performance of FLAN-T5 and GPT-3.5 on both single-hop and multi-hop tasks. It's surprising that chain-of-thought prompting appears to negatively impact the performance of GPT-3.5. However, it proves beneficial for GPT-3.5. in the multi-hop task. We speculate that the effectiveness of chain-of-thought prompting increases when the task is more challenging.

Model	Pragmatic	Visual	Combined
FLAN-T5-XXL	93.04	82.91	79.75
LLaMA2	86.71	70.25	69.62
LLaVA	92.41	74.05	73.25
GPT-3.5	91.14	82.28	79.75
GPT-4	95.57	88.61	85.44

Table 4: **Results on pragmatic reasoning, visual knowledge reasoning, and our original data (combined).** We study different types of reasoning in our data. We report individual accuracy.

3.4 Results of Encoder-only Models

Encoder-only models cannot be applied to our task without fine-tuning. Thus, we fine-tune the encoder-only models on natural language inference datasets instead. By doing this, our task is framed into the NLI setup and choose an option by the highest probability of the 'entailment' class. We fine-tune the encoder-only models with SNLI (Bowman et al., 2015) and ANLI (Nie et al., 2019) datasets and we use only 'contradiction' and 'entailment' labels. Table 3 shows the results of encoder-only models. VL-BERT and Oscar are BERT-based vision-language models, and they are trained on image-caption datasets. In our experiments, we observe that the vision-language models consistently surpass the BERT model on our dataset.

3.5 Pragmatic and Visual Knowledge Reasoning

We investigate whether models genuinely understand visual knowledge for our task. Our task requires pragmatic reasoning and visual knowledge reasoning. We decouple our task into pragmatic reasoning and visual knowledge reasoning and analyze which step is a bottleneck. Table 4 shows the results on pragmatic reasoning (pragmatic), visual knowledge reasoning (visual), and our original data (combined), utilizing the same subset. Firstly, results on pragmatic reasoning are better than others, suggesting that large models do well on pragmatic reasoning. For example, GPT-4 achieves 95.57% on pragmatic reasoning. Main bottleneck in our task is on visual knowledge reasoning; results on visual knowledge reasoning are lower than those on pragmatic reasoning. When comparing LLaMA2 and LLaVA, LLaVA demonstrates superior abilities in both pragmatic reasoning and visual knowledge reasoning. Interestingly, FLAN-T5-XXL performs comparably to a proprietary model, GPT-3.5, in terms of pragmatic reasoning and visual reasoning.

4 Conclusion

Examining real-world object properties requires a visual understanding that language models lack. In our study, we introduced a text-only WINOVIZ focused on question-answering tasks, comprising 1,380 examples exploring language models' reasoning capabilities across various visual properties of objects in diverse contexts. Our findings revealed that large language models demonstrate effective performance overall but struggle particularly with the multi-hop version of our dataset. It became apparent that the bottleneck in our task lies in the reasoning of visual knowledge. Vision-language models surpass their language-only counterparts, although image-generation approaches prove ineffective for our specific task. Future endeavors will delve into how to efficiently transfer visual knowledge from images or captions.

5 Limitations

Our work is focused on a specific subset of language models and vision-language models. We adopt vision-language models in which the language backbones are pre-trained using image-caption datasets. Additionally, we employ Stable Diffusion for image generation, although the current output may not directly benefit our task. Utilizing state-of-the-art diffusion models could enhance image quality, yet the challenge of generating images useful for our task persists. Moreover, our observations indicate that large language models excel in our single-hop task, achieving up to 90% accuracy. This suggests that these large models can effectively reason over visual knowledge even in the absence of explicit visual signals. Nonetheless, how visual signals can be harnessed to enhance language models is underexplored, and we defer it to future research endeavors.

References

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.

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*.

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. *Advances in neural information processing systems*, 33:1877–1901.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Kevin Crowston. 2012. Amazon mechanical turk: A research tool for organizations and information systems scholars. In *Shaping the Future of ICT Research. Methods and Approaches: IFIP WG 8.2, Working Conference, Tampa, FL, USA, December 13-14, 2012. Proceedings*, pages 210–221. Springer.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven C. H. Hoi. 2023. Instructblip: Towards general-purpose vision-language models with instruction tuning. *CoRR*, abs/2305.06500.

Yuling Gu, Bhavana Dalvi Mishra, and Peter Clark. 2022. Do language models have coherent mental models of everyday things? *arXiv preprint arXiv:2212.10029*.

Lovisa Hagström and Richard Johansson. 2022. What do models learn from training on more than text? measuring visual commonsense knowledge. *arXiv preprint arXiv:2205.07065*.

Woojeong Jin, Dong-Ho Lee, Chenguang Zhu, Jay Pujara, and Xiang Ren. 2022. Leveraging visual knowledge in language tasks: An empirical study on intermediate pre-training for cross-modal knowledge transfer. *arXiv preprint arXiv:2203.07519*.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023a. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*.

Lei Li, Jingjing Xu, Qingxiu Dong, Ce Zheng, Qi Liu, Lingpeng Kong, and Xu Sun. 2023b. Can language models understand physical concepts? *arXiv preprint arXiv:2305.14057*.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX 16*, pages 121–137. 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 *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*.

Xiao Liu, Da Yin, Yansong Feng, and Dongyan Zhao. 2022. Things not written in text: Exploring spatial commonsense from visual signals. *arXiv preprint arXiv:2203.08075*.

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.

Ken McRae, George S Cree, Mark S Seidenberg, and Chris McNorgan. 2005. Semantic feature production norms for a large set of living and nonliving things. *Behavior research methods*, 37(4):547.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*.

Tobias Norlund, Lovisa Hagström, and Richard Johansson. 2021. Transferring knowledge from vision to language: How to achieve it and how to measure it? *arXiv preprint arXiv:2109.11321*.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Cory Paik, Stéphane Aroca-Ouellette, Alessandro Roncone, and Katharina Kann. 2021. The world of an octopus: How reporting bias influences a language model's perception of color. *arXiv preprint arXiv:2110.08182*.

Ehsan Qasemi, Filip Ilievski, Muhao Chen, and Pedro Szekely. 2021. Paco: Preconditions attributed to commonsense knowledge. *arXiv preprint arXiv:2104.08712*.

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. In *International conference on machine learning*, pages 8748–8763. PMLR.

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. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695.

Shikhar Singh, Ehsan Qasemi, and Muhao Chen. 2022. Viphy: Probing" visible" physical commonsense knowledge. *arXiv preprint arXiv:2209.07000*.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Hao Tan and Mohit Bansal. 2020. Vokenization: Improving language understanding with contextualized, visual-grounded supervision. *arXiv preprint arXiv:2010.06775*.

Zineng Tang, Jaemin Cho, Hao Tan, and Mohit Bansal. 2021. Vidlankd: Improving language understanding via video-distilled knowledge transfer. *Advances in Neural Information Processing Systems*, 34:24468–24481.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Chenyu Zhang, Benjamin Van Durme, Zhuowan Li, and Elias Stengel-Eskin. 2022. Visual commonsense in pretrained unimodal and multimodal models. *arXiv preprint arXiv:2205.01850*.

Wanrong Zhu, An Yan, Yujie Lu, Wenda Xu, Xin Eric Wang, Miguel Eckstein, and William Yang Wang. 2022. Visualize before you write: Imagination-guided open-ended text generation. *arXiv preprint arXiv:2210.03765*.

A Appendix

A.1 Data Collection

The data collection is broken down into three sections: (1) collecting candidate objects, (2) annotating premise and hypothesis sentences, (3) verifying the quality of the annotated dataset, and (4) human evaluation.

Object Collection. To begin with, we gather a collection of objects along with their potential properties or attributes for constructing our data. These objects and attributes are obtained by scraping information from reliable sources such as Memory Colors (Norlund et al., 2021), Visual Property Norms (Hagström and Johansson, 2022), and McRae feature norms (McRae et al., 2005). Through this process, we manage to collect a total of 800 unique objects and 302 unique attributes. However, it is necessary to refine our dataset by filtering out attributes that are either too abstract or non-visual in nature. To accomplish this, we employ specific heuristics to ensure the inclusion of only concrete and visually relevant attributes. As a result of this filtering process, we successfully obtain a final dataset comprising 775 objects and 156 attributes.

Dataset Annotation. We utilized Amazon Mechanical Turk (Crowston, 2012) for data annotation, as depicted in Figure 1. The data annotation process involves several steps. Initially, annotators are given an object, and are instructed to identify two properties for the object and corresponding visual attributes for those properties. For example, for the object *banana*, the annotator may come up with two properties *ripe* and *rotten*, which would have corresponding visual attributes *yellow* and *brown*, respectively. After identifying a pair of object properties and visual attributes, they are tasked with composing natural language sentences for each attribute and property. The properties are associated with premise sentences, while the attributes were linked to hypothesis sentences.

Annotators were selected from a small pool of Mechanical Turkers that the authors had previously worked with. The Turkers had to further pass a qualification task that tested their understanding of the annotation task. The authors manually examined the annotations to ensure quality of the collected data.

Model	# Params	Public	VL model
BERT-Base	109M	✓	✗
BERT-Large	335M	✓	✗
VL-BERT-Large	335M	✓	✓
Oscar-Large	335M	✓	✓
CLIP-Large	427M	✓	✓
FLAN-T5-XXL	11B	✓	✗
InstructBLIP	11B	✓	✓
LLaMA2	13B	✓	✗
LLaVA	13B	✓	✓
GPT-3	175B	✗	✗
GPT-3.5	Unknown	✗	✗
GPT-4	Unknown	✗	✗

Table 5: **A list of models used in the experiments:** BERT (Kenton and Toutanova, 2019), CLIP (Radford et al., 2021), VL-BERT (Su et al., 2019), Oscar (Li et al., 2020), FLAN-T5 (Chung et al., 2022), Instruct-BLIP (Dai et al., 2023), LLaMA2 (Touvron et al., 2023), LLaVA (Liu et al., 2023), GPT-3 (Brown et al., 2020; Ouyang et al., 2022), and GPT-4 (OpenAI, 2023). We use the 'text-davinci-003' API for GPT-3, 'gpt-3.5-turbo-instruct' for GPT-3.5, and 'gpt-4-0314' for GPT-4.

Model	Ind.	Pair
FLAN-T5-Base (No imgs)	67.89	40.37
CLIP-Large	64.67	36.46
FLAN-T5-XXL (No imgs)	86.24	72.71
FLAN-T5-XXL (Captions)	85.83	71.88
InstructBLIP	53.21	22.93

Table 6: **Results on WINOVIZ with generated images.** We use Stable Diffusion (Rombach et al., 2022) to generate 5 images per premise sentence. We adopt majority voting at inference time to choose an option. FLAN-T5-Base (No imgs) refers to a model without any generated images, with a size comparable to CLIP-Large. FLAN-T5-XXL (No imgs) refers to a model without any generated images, while FLAN-T5-XXL (Captions) refers to a model with captions generated by BLIP2 on the generated images. Instead of directly inputting images into FLAN-T5, we extract captions from the generated images and use them as additional context. InstructBLIP uses generated images.

A.2 Versions of WINOVIZ

We now collect our WINOVIZ data. We also propose the multi-hop data, a more challenging version of WINOVIZ, and a dataset for probing visual knowledge. For the multi-hop data, we create new hypothesis options that require more intermediate steps while we simplify the premise sentences to measure the ability of models about visual knowledge.

Multi-hop Data. To create a more challenging task, we introduce a multi-hop version of our data, which requires more intermediate steps. The basic idea of the multi-hop data is to replace a visual attribute word in hypotheses with another object word which has a similar visual attribute. This requires one more reasoning step to find out the visual attribute. For example, one hypothesis option is 'The banana is yellow.'. Then 'yellow' can be replaced with 'the color of an egg yolk.' So the new hypothesis option for the multi-hop version is 'The banana is the color of an egg yolk.' The multi-hop version is more challenging since a model has to find out what color is an egg yolk. We focus on color, shape, material on the multi-hop data and curate prototypical objects for each visual property word. We get 200 samples for the multi-hop data.

Pragmatic Reasoning vs. Visual Knowledge Reasoning. Another important aspect of this work is that models genuinely understand and know visual knowledge. Our task requires pragmatic reasoning, the process of finding the intended meaning, and visual knowledge reasoning but models may fail in one of the reasoning steps. Thus, we decouple the premise sentence into pragmatic reasoning step and visual knowledge reasoning step to analyze which step is a bottleneck. Pragmatic reasoning involves finding the intended meaning and finding key phrases for the next step, visual knowledge reasoning. For example, a model should first find 'the banana is ripe' given the premise sentence in the pragmatic reasoning step (Figure 1). Given the simplified sentence, a model should choose a better option, 'the banana is yellow', in the visual knowledge reasoning step. We obtain 160 samples to study this (Section 3.5).

A.3 Using Image Generation for WINOVIZ Task.

Another approach for our task is to utilize image generation. We generate images based on premise sentences and employ these generated images for our task. The generated images may contain useful information that assists in identifying a correct hypothesis. We utilize an image generation approach, Stable Diffusion (Rombach et al., 2022), to generate images. We use the generated images to guide the LMs inspired by imagination-guided text generation (Zhu et al., 2022). Given the generated images, there are three ways to use them. The first method involves using CLIP (Radford et al., 2021) on both the images and hypothesis sentences to identify a superior hypothesis option. Specifically, we calculate the cosine similarity between the embedding of a generated image and the embedding of a hypothesis option, selecting the hypothesis with a higher cosine similarity score. The second approach is to generate captions for the generated images using a caption model. Since language models cannot directly process images, we generate captions and utilize them as additional context for the task. BLIP2 (Li et al., 2023a) is employed for caption generation. The third strategy is to reframe our task as a visual question-answering task and employ a vision-language model to identify a better option. In this setup, we use InstructBLIP (Dai et al., 2023). For image generation, we use Stable Diffusion (Rombach et al., 2022), generating 5 images per premise sentence. A better hypothesis option is determined through majority voting.

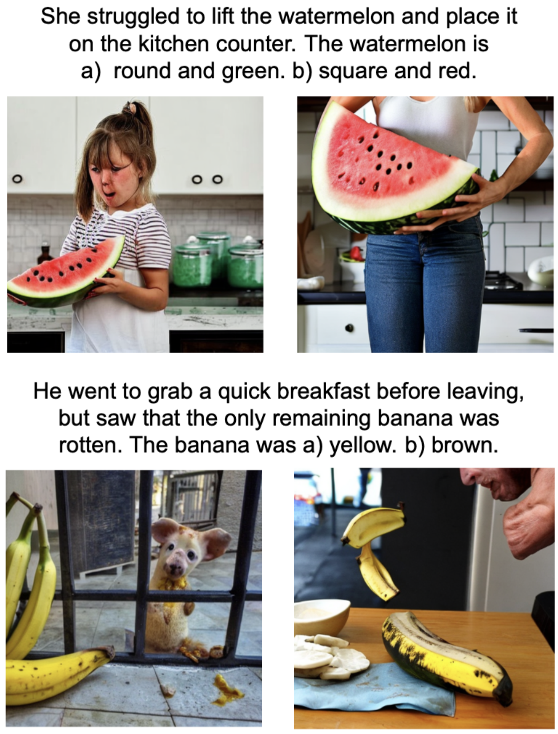


Figure 3: **Examples of generated images.** We generate images using Stable Diffusion (Rombach et al., 2022). In the second example, the bananas in both images are yellow, leading the model to select the incorrect option. The generated image examples don't assist in selecting a more plausible hypothesis option.

Table 6 displays the outcomes related to image generation. The first approach utilizing CLIP falls short compared to FLAN-T5-Base which is slightly smaller than CLIP-Large. In the second approach involving BLIP2 captions, we opt for FLAN-T5-XXL as the benchmark, comparing one scenario with no additional data and another incorporating captions from generated images. Our experiment reveals a notable decline in performance when captions are employed. The third approach significantly underperforms FLAN-T5-XXL by a large margin. These experiments collectively indicate that generated images offer limited utility for our task. Furthermore, a manual assessment of 100 generated images reveals that 66% of them do not contribute meaningfully to our objectives. Examples of generated images with premise sentences are shown in Figure 3. In the figure, the bananas in both images are yellow; the generated images do not provide any clues to choose a more plausible option.

A.4 Related Work

There are multiple perspectives on how our contributions relate to previous work, and we elaborate on this in the subsequent sections.

Visual Knowledge Probing. Several attempts have been made to assess the reasoning ability of language models regarding objects, primarily through natural language benchmarks (Norlund et al., 2021; Hagström and Johansson, 2022; Paik et al., 2021; Zhang et al., 2022; Singh et al., 2022; Qasemi et al., 2021). Norlund et al. (2021) introduced a task involving querying a multimodal model for visual commonsense knowledge related to memory colors, which are the typical colors associated with well-known objects. Hagström and Johansson (2022) expanded on this work by proposing visual property norms as a measure of visual commonsense knowledge in both language models and multimodal models. Paik et al. (2021) evaluated the color perception of language models using a color dataset called CoDa, revealing that reporting bias negatively affects model performance and that multimodal training can alleviate these effects. Zhang et al. (2022) confirmed these findings and extended the evaluation to a wider range of visually salient properties. Similarly, Singh et al. (2022) evaluated vision-language models on a visually accessible commonsense knowledge dataset. Liu et al. (2022) explored spatial commonsense, the knowledge about spatial position and relationship between objects, finding that image synthesis models are more capable of learning accurate and consistent spatial knowledge than other models. Gu et al. (2022) proposed a probing dataset for physical knowledge about everyday things. In contrast, we present a challenging dataset that probes the reasoning abilities of language models regarding variant visual properties of objects under different context.

Vision-Language Modeling Recent advances in vision-language (VL) models have led to success on vision-language tasks such as visual question answering, captioning, and grounding (Antol et al., 2015; Lin et al., 2014; Mao et al., 2016). Existing VL models jointly learn image and text representations through cross-modal alignments including VL-BERT (Su et al., 2019), LXMERT (Tan and Bansal, 2019), Oscar (Li et al., 2020). Recent approaches have introduced visual instruction tuning, which involves fine-tuning a VL model using instruction-following data (Liu et al., 2023).

While these VL models have shown significant improvement in VL tasks, the exploration of how to transfer visual knowledge from VL modeling to language tasks remains underexplored. Vokenization (Tan and Bansal, 2020) utilized token-level text-to-image retrieval to transfer visual knowledge to language models. VidLanKD (Tang et al., 2021) employd contrastive learning to train a teacher model on video datasets and uses distillation approaches to transfer visual knowledge from the teacher to a student model. CMKT (Jin et al., 2022) investigated two types of knowledge transfer: text knowledge transfer (e.g., captions) and visual knowledge transfer (e.g., images and captions). Their findings demonstrate that such transfer can enhance performance on commonsense reasoning tasks.

A.5 Annotation Interfaces

We provide Turking interfaces: qualification task in Figure 4, and annotation task in Figures 5, 6, 7, 8.

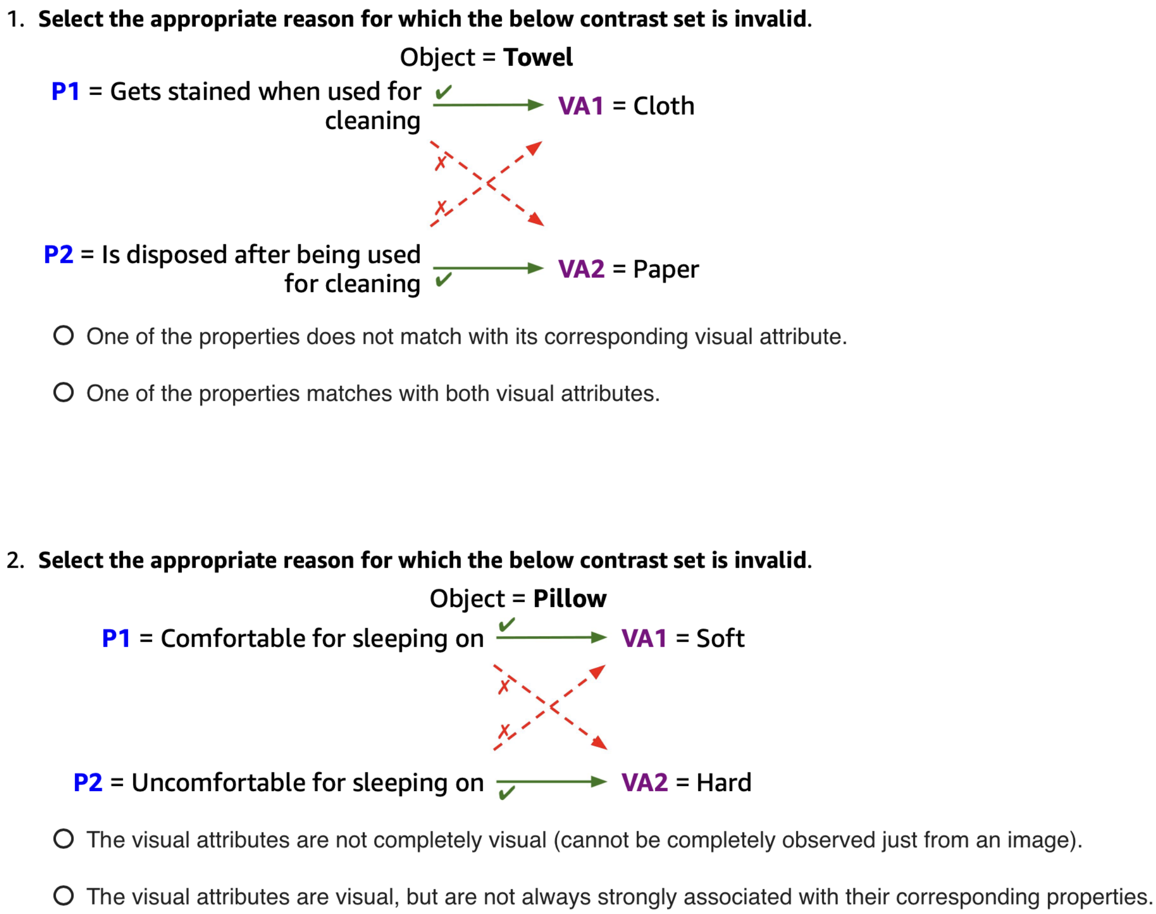
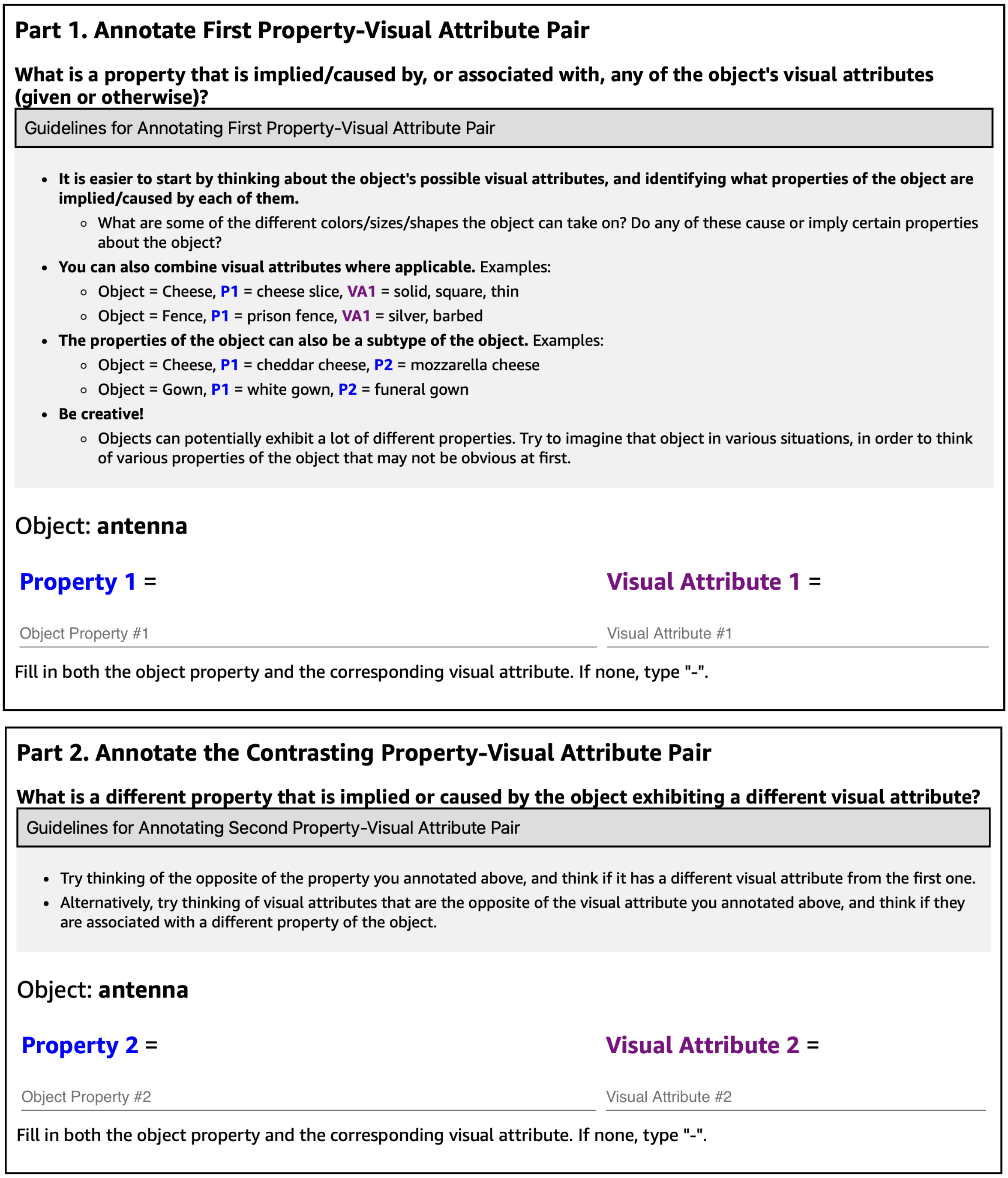


Figure 4: **The Interface of the qualification task.** We provide 12 questions to find quality workers.



Part 1. Annotate First Property-Visual Attribute Pair

What is a property that is implied/caused by, or associated with, any of the object's visual attributes (given or otherwise)?

Guidelines for Annotating First Property-Visual Attribute Pair

- **It is easier to start by thinking about the object's possible visual attributes, and identifying what properties of the object are implied/caused by each of them.**
  - What are some of the different colors/sizes/shapes the object can take on? Do any of these cause or imply certain properties about the object?
- **You can also combine visual attributes where applicable.** Examples:
  - Object = Cheese, **P1** = cheese slice, **VA1** = solid, square, thin
  - Object = Fence, **P1** = prison fence, **VA1** = silver, barbed
- **The properties of the object can also be a subtype of the object.** Examples:
  - Object = Cheese, **P1** = cheddar cheese, **P2** = mozzarella cheese
  - Object = Gown, **P1** = white gown, **P2** = funeral gown
- **Be creative!**
  - Objects can potentially exhibit a lot of different properties. Try to imagine that object in various situations, in order to think of various properties of the object that may not be obvious at first.

Object: **antenna**

Property 1 = **Visual Attribute 1** =

Object Property #1 Visual Attribute #1

Fill in both the object property and the corresponding visual attribute. If none, type "-".

Part 2. Annotate the Contrasting Property-Visual Attribute Pair

What is a different property that is implied or caused by the object exhibiting a different visual attribute?

Guidelines for Annotating Second Property-Visual Attribute Pair

- Try thinking of the opposite of the property you annotated above, and think if it has a different visual attribute from the first one.
- Alternatively, try thinking of visual attributes that are the opposite of the visual attribute you annotated above, and think if they are associated with a different property of the object.

Object: **antenna**

Property 2 = **Visual Attribute 2** =

Object Property #2 Visual Attribute #2

Fill in both the object property and the corresponding visual attribute. If none, type "-".

Figure 5: **Interfaces of annotating visual contrast sets (parts 1 and 2).**

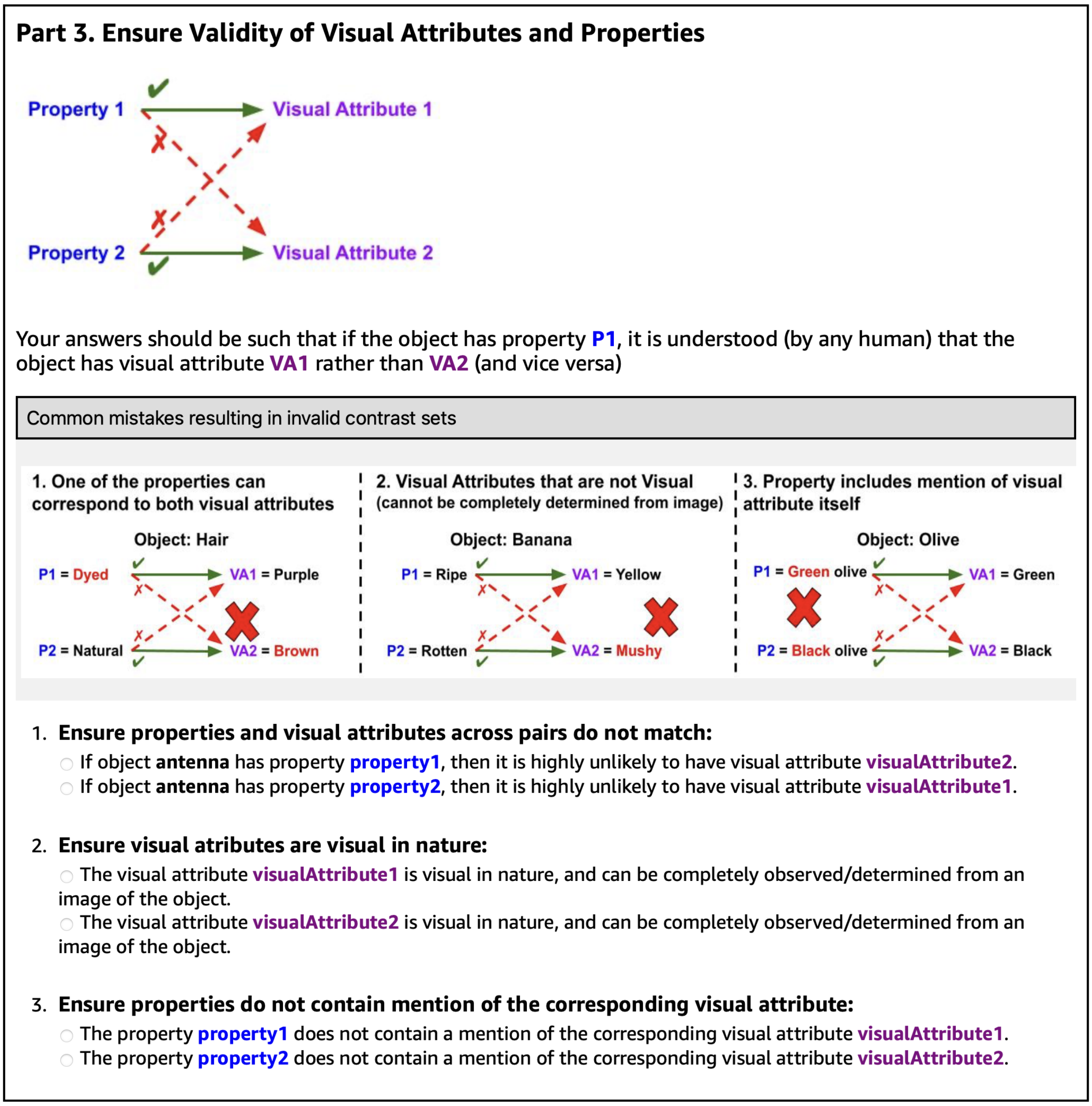


Figure 6: **Interfaces of annotating visual contrast sets (part 3).**

Part 1: Create Sentences about the Object Properties

For each of the two properties you annotated above, write a sentence about the object in a real-world situation where it is exhibiting that property.

- The sentence should specifically mention both the object, and its property. **It should be clear from the sentence that the object in this sentence has that property**.
- The sentence should **NOT** mention the visual attribute corresponding to this property.
- Sentence should be **at least 10 words long**, and grammatically correct (begin with capital letter, end with punctuation).
- **Make the sentence as free-form and creative as possible.** Involve one or more characters in the sentence if possible.
- **DO NOT** make very short and simple sentences, that directly mention the object having a property and nothing else.Examples of bad sentences:
  - Object = **plate**, Property = **folded**, Sentence = *The plate was folded.*
  - Object = **cinnamon**, Property = **ground**, Sentence = *The cinnamon was ground.*
- If the two properties are opposites of each other (e.g. ripe vs rotten), you can write near-identical sentences with just the properties switched. However, do not force this if it does not make sense for the selected properties.

Example Sentences about Object Properties

- Object = **banana**, Property = **rotten**. Example Sentence: *He went to grab a quick breakfast before leaving, but saw that the only remaining banana was rotten.*
- Object = **box**, Property = **can be carried in one hand**. Example Sentence: *She already had the carpet in one hand, but picked up another box before heading up to the apartment.*
- Object = **cheese**, Property = **cheese slice**. Example Sentence: *He added a slice of cheese to his turkey sandwich.*
- Object = **napkin**, Property = **worn on lap at restaurants**. Example Sentence: *She placed the napkin across her lap before starting to eat her dinner.*

Object = antenna

Property 1 =

Property Sentence #1

Property 2 =

Property Sentence #2

Part 2: Create Sentences about the Visual Attributes

For each of the two visual attributes you annotated above, write a simple sentence that explicitly states that the object has that visual attribute.

- The sentence should specifically mention both the object and its visual attribute, and nothing else.
- The sentence should **NOT** mention the property corresponding to that visual attribute.
- Make the sentence as simple as possible. For e.g., "The banana was yellow", "The cheese was solid", "The nail was made of metal".
- Make the two sentences as identical to each other as possible, with only the visual attribute being different.
- Match the tense of the sentence to the tense of the corresponding property sentence - if the property sentence is in past tense, make the visual attribute sentence in past tense as well.

Example Sentences about Visual Attributes

- Object = **banana**, Visual Attribute = **black**. Example Sentence: *The banana was black.*
- Object = **box**, Visual Attribute = **small**. Example Sentence: *The box was small.*
- Object = **cheese**, Visual Attribute = **solid, square**. Example Sentence: *The cheese was solid and square.*

Suggested sentence format: *The **OBJECT** is/was **VISUAL ATTRIBUTE**.*

Object = antenna

Visual Attribute 1 =

Visual Attribute Sentence #1

Visual Attribute 1 =

Visual Attribute Sentence #2

Figure 7: **Interfaces of converting contrast sets into sentence puzzles (parts 1 and 2).**

Part 3: Ensure Validity of Final Puzzle

Solve the puzzle you've created! Ensure that for each object property sentence, the sentence about the corresponding visual attribute is more likely to be true.

Puzzle Part 1:

Property Sentence:
Which of the choices is more likely to be true?

○

○

Puzzle Part 2:

Property Sentence:
Which of the choices is more likely to be true?

○

○

Grammatical Correctness:

The four sentences you created are:

1. **Property Sentence 1:**
2. **Property Sentence 2:**
3. **Visual Attribute Choice 1:**
4. **Visual Attribute Choice 2:**

○ All four sentences above are grammatically correct.

○ All four sentences are properly capitalized, and begin with capital letters.

○ All four sentences end in punctuation.

Figure 8: **Interfaces of converting contrast sets into sentence puzzles (part 3).**

Harnessing the Power of Multiple Minds: Lessons Learned from LLM Routing

KV Aditya Srivatsa* **Kaushal Kumar Maurya*** **Ekaterina Kochmar**
Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE
{vaibhav.kuchibhotla, kaushal.maurya, ekaterina.kochmar}@mbzuai.ac.ae

Abstract

With the rapid development of LLMs, it is natural to ask how to harness their capabilities efficiently. In this paper, we explore whether it is feasible to direct each input query to a single most suitable LLM. To this end, we propose *LLM routing* for challenging reasoning tasks. Our extensive experiments suggest that such routing shows promise but is not feasible in all scenarios, so more robust approaches should be investigated to fill this gap.[1]

1 Introduction

Large language models (LLMs) demonstrate remarkable capabilities in many natural language generation and understanding tasks (Bommasani et al., 2021; Chang et al., 2023; Minaee et al., 2024). At the same time, Jiang et al. (2023) show that no single open-source LLM outperforms all others across different benchmarks and datasets, as various LLMs may exhibit different domain expertise (Beeching et al., 2023). Experiments towards predicting model behavior (Rabinovich et al., 2023; Srivatsa and Kochmar, 2024) also suggest that particular aspects of input prompts can affect different LLMs in different ways.

It is, therefore, reasonable to investigate whether the capabilities of different LLMs can be harnessed to achieve better results more efficiently. Recent findings suggest that performance can be improved with ensembling (Wang et al., 2022, 2023; Li et al., 2024) and collaborative frameworks (Wu et al., 2023b; Li et al., 2023). However, the research in this area is still in the early stages, with a number of open research questions. In this work, we propose *LLM routing*, which investigates whether *directing an input prompt to the most suitable single LLM can lead to better performance than what can be achieved with individual LLMs while maintaining a reasonable (e.g., single LLM) latency*.

With the rise of larger and more capable models in NLP and the wider field of ML, the research on sparse expert models has also extended. This class of models includes mixture-of-experts (Jacobs et al., 1991; Collobert et al., 2002; Eigen et al., 2013), switch-transformers (Fedus et al., 2022), and routing networks (Rosenbaum et al., 2017), among other models.[2] Approaches to building these sparse models vary along several dimensions: (i) how the optimal parameter subset(s) or model-pool candidates are identified for each input (e.g., feature-based or deep-encoder-based classification), (ii) whether the subset selection involves pre-training the candidate models or model layers (e.g., Mixtral (Jiang et al., 2024)), which can incur significant training compute and data costs, (iii) how many experts are selected for each input (e.g., HybridLLM (Ding et al., 2024) selects only the single best, whereas Shazeer et al. (2017) selects the top-k), and (iv) whether the approach also aims to improve the response quality or overall performance beyond that of any single candidate model. In this context, our paper aims to build and analyze a sparse LLM routing model that selects the single best LLM (from a pool of at least two LLMs) for each input query. The proposed router only requires fine-tuning of a relatively small pre-trained Transformer encoder model on the data without the need for pre-training or fine-tuning the LLMs.

Given that LLMs frequently face challenges with reasoning and planning tasks (Wei et al., 2022; Kojima et al., 2022), we focus on two well-established reasoning task benchmarks. We empirically investigate the feasibility of building *LLM routing* model capable of selecting the most suitable LLM for each input from a pool of diverse LLMs. The routing is grounded on responses generated by LLMs.

*Equal contribution

[1]Our code and data are available at https://github.com/kvadityasrivatsa/llm-routing.

[2]For more details on the related work, see Appendix F.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 124–134
June 20, 2024

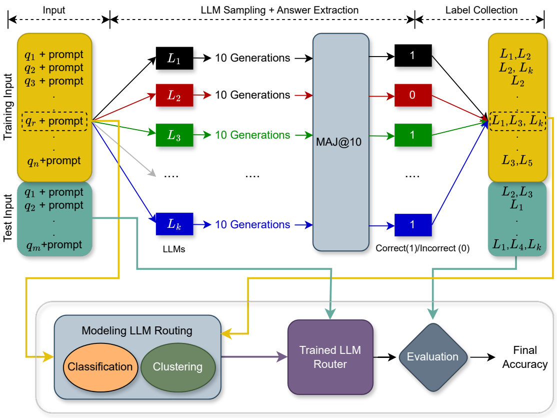


Figure 1: Overview of the proposed workflow.

We explore binary and multi-label classification modeling at the input query level, as well as a clustering approach based on the similarity between the queries. Finally, leveraging prediction confidence scores, we design multiple optimal policies to select a single suitable LLM from the pool.

The contributions and key findings of this work are as follows: (1) We propose an LLM routing model, which directs input queries to the most suitable *single* LLM. (2) We explore two different types of approaches for LLM routing, treating it as a classification and a clustering task. (3) We conduct experiments with 7 open-source LLMs and on two challenging reasoning benchmarks (`GSM8K` and `MMLU`). (4) We introduce theoretical upper bounds for two scenarios: (i) highest possible performance achieved jointly with all LLMs (i.e., oracle), and (ii) highest performance achieved with the proposed routing model. (5) Our findings indicate that theoretical upper bounds of the routing model are higher than individual model performance, however, the proposed model developed in practice is unable to achieve those scores. Specifically, the performance of the routing model is better than that of the weak LLMs but is similar to or slightly lower than that of the top-performing LLMs, which may be due to the small size of the training data.

Despite the somewhat negative results, we believe this study demonstrates the feasibility of modeling LLM routing and contributes to new research directions on efficient usage of LLMs, which can benefit researchers and practitioners.

2 Methodology

We present an overview of the proposed workflow in Figure 1. Below, we describe our approaches to *LLM sampling* and *LLM routing*.

Split/Criteria	GSM8K	MMLU
Training	6,816	13,757
Validation	359	285
Test	1,319	1,530
#examples for few-shot CoT	5	5

Table 1: Dataset statistics for the `GSM8K` and `MMLU` datasets. `MMLU` data splits are remapped to have a distribution similar to `GSM8K`. `CoT`: Chain-of-Thought

LLMs	Chat?	Specialized?	#Parameters
`llama2-7b`	×	×	7B
`llama2-13b-chat`		×	13B
`mistral-7b`	×	×	7B
`mistral-7b-it`		×	7B
`gemma-7b`	×	×	7B
`gemma-7b-it`		×	7B
`metamath-7b`	×		7B

Table 2: List of diverse LLMs selected in this study.

2.1 LLM Sampling

Selection of Benchmarks and LLMs As it has been observed that most of the existing LLMs struggle with reasoning tasks (Patel et al., 2021; Wu et al., 2023a), we focus on two challenging datasets associated with distinct domains – mathematical (`GSM8K` by Cobbe et al. (2021)) and natural language reasoning (`MMLU` by Hendrycks et al. (2021)). `GSM8K` consists of 8,792 diverse grade-school level math word problems (MWPs), while `MMLU` contains 15k multiple-choice questions spanning 57 subjects across STEM, humanities, and social sciences, among others (see Table 1). We have selected diverse LLMs based on criteria such as performance on benchmarks, training methodologies, model specialization, and more. The final set of LLMs is presented in Table 2.

Routing Data In this study, we assess each LLM's performance by generating 10 responses for each input query to ensure more reliable and replicable behavior in our modeling. For LLM prompting and answer extraction from responses, we have followed the standard guidelines (see Appendices B and C for details). Figure 2 presents the sample prompting templates. We use majority voting scores as labels for each input query to train routing classifiers. *Majority Voting* (MAJ@K $\in \{0, 1\}$) determines whether the most frequent answer matches the gold answer or not. The mean MAJ@10 scores across all input queries are reported in Table 3. Furthermore, to ensure a reliable response from an LLM, we consider only those LLMs for which the extracted answer viability scores are above 90% (please refer to Appendix B for more details), resulting in 6 viable LLMs for the GSM8K dataset and 7 for the MMLU dataset, respectively. We prepare the routing dataset by associating each input query with those viable LLM(s) that have a MAJ@10 score of 1. Formally, the target label for an input query $q \in Q$ is given by $label\,(q) = \{l \mid l \in L, maj@10\,(q, l) = 1\}$, where L is the set of candidate LLMs and Q is the set of query prompts from GSM8K or MMLU.

2.2 LLM Routing

Next, we build an LLM router, *determining which model to select from a pool of LLMs for a given input query based on performance and inference latency*. The ideal routing algorithm should select an optimal single LLM with high accuracy and low latency. To this end, we explore modeling at the individual query level using classification and utilize similarities among queries using clustering.

Classifier-Based Routing The classification-based routing consists of (1) the development of a classifier that can predict a set of LLMs capable of solving the input query along with prediction confidence scores, and (2) the identification of the policy to select optimal LLMs (with high accuracy and low latency) from the predicted LLMs based on confidence scores in the range [0-1].

Multi-label and Separate Classifiers: We have considered two types of classifiers: a multi-label classifier (MLC) and separate classifiers (SC). MLC aims to predict all LLMs apt for a given input query together in a single prediction step. The SC model, on the other hand, employs a separate binary classifier for each LLM and accumulates the results post hoc. Both types of classifiers are built on top of existing popular pre-trained language models (PLMs). Specifically, we experimented with BERT, DistilBERT, RoBERTa, and T5 models. Additionally, due to the small size of the training data, we explored smaller models, utilizing only a few layers of PLMs, as well as simpler models such as Random Forests. RoBERTa emerged as the best-performing model, and all results in this paper are reported with classifiers built by fine-tuning the RoBERTa PLM exclusively.

Proposed Policies: We utilize the classifiers' predicted confidence scores to design the following policies:

1. **ArgMax:** Select an LLM with the highest confidence score.
2. **Random:** Select a pool of LLMs with confidence above a certain threshold (i.e., 0.80) and randomly pick one LLM from the pool.
3. **Prediction:** Train a RandomForest regressor using training data confidence scores, where each input represents the confidence score for each predicted label, and the target is the confidence score of the first gold reference LLM. At test time, we select the LLM with a confidence score closest to the predicted score.
4. **Sorted Prediction (Sorted Pred):** Similarly to the 'Prediction' policy, the input confidence scores are arranged in ascending order based on LLMs' performance. This ensures that weaker LLMs have a fair opportunity.

Clustering-Based Routing Additionally, to incorporate the query-level similarities, we explore clustering for LLM routing as detailed below.

Learning Clusters: We fit a KMeans[3] clustering model on query-specific features extracted from the training data to learn discrete clusters. The features are extracted using: (1) TF-IDF vectorizer,[4] and (2) pooled hidden embedding of the RoBERTa[5] model's last layer.

Routing: For each cluster in the training set, the best performing LLM is determined. At inference, input queries in the test set are routed to the best-performing LLM for their corresponding cluster.

3 Experimental Setup

LLM Routing Baseline Models The following baseline models are included for comparison:

1. **Oracle:** The maximum possible performance is assumed under the premise that an oracle always selects a single LLM capable of solving each query if it is solvable.
2. **Random:** This represents the mean performance of randomly selecting an LLM uniformly for each input query across 1000 independent runs.
3. **Individual Models:** This is the mean performance of individual models with MAJ@10 across all queries.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[5] https://huggingface.co/FacebookAI/roberta-base

Models		GSM8K		MMLU	
		Acc	Lat (sec)	Acc	Lat (sec)
Oracle		87.18	3.46	89.15	1.89
Random		55.37	3.52	52.50	2.35
`gemma-7b`		<u>71.11</u>	7.10	<u>63.85</u>	3.00
`metamath-7b`		67.55	4.70	42.28	2.40
`mistral-7b`		59.74	3.70	62.09	1.80
`*mistral-7b-it`		50.41	1.00	51.63	1.10
`llama2-13b-chat`		46.70	1.80	50.52	4.80
`*gemma-7b-it`		36.84	0.70	49.28	1.00
`llama2-7b`		–	–	48.36	2.30
All LLMs		74.37	19.00	60.39	16.40
MLC	Upper bound	79.68	5.16	77.18	1.94
	ArgMax policy	67.62	4.76	62.28	2.95
	Random policy	67.47	4.76	58.16	2.86
	Prediction policy	**67.70**	4.77	**63.85**	2.95
	Sorted Pred policy	59.90	4.77	48.36	2.92
SC	ArgMax policy	67.55	4.70	62.87	2.94
Clustering	TF-IDF	67.55	4.70	61.76	2.83
	RoBERTa	67.55	4.70	61.76	2.83

Table 3: Performance of different routing models on `GSM8K` and `MMLU` test sets. For all queries, we have considered 10 generations with each LLM. Acc: mean accuracy with MAJ@10 (%), Lat: LLM inference latency in seconds per query (10 generations for each query), MLC: multi-label classifier, and SC: separate classifiers. * The term '`it`' indicates instruction-tuned LLMs. The highest individual-LLM accuracy is <u>underlined</u>, and the highest classifier accuracy is in **bold** for each dataset.

4. **All LLMs:** This baseline reports the mean accuracy of MAJ@$(10\times|L|)$ based on the combined pool of 10 generations from each LLM, where $|L|$ is the total number of LLMs.

Classifier Upper Bound This is similar to the oracle model, where the upper bound is calculated with predicted labels instead of gold labels.

4 Results and Discussion

In Table 3, we present the performance of each individual LLM across both datasets, alongside the performance of baselines and routing models. We observe that, even though `gemma-7b` outperforms other LLMs, there are diverse performance trends for other LLMs across datasets, with some performing better on `GSM8K`, and others on `MMLU`. To investigate the results further, we pose and address a number of research questions.

Does including multiple LLMs solve all questions in a given dataset? The Oracle model's Acc scores for both datasets are lower than 90%, indicating that more than 10% of questions cannot be solved by all LLMs combined. For details, see Figure 3 in the Appendix, where we project the distribution of questions solved by each of the LLMs.

How effective is a routing model when randomly picking LLMs? As expected, the random baseline model achieves the lowest Acc score for both datasets. This highlights the necessity for an effective routing model to navigate through LLMs.

Is the joint performance of multiple LLMs better than that of individual LLMs? Considering extreme cases like top-k and bottom-k LLMs as shown in Appendix Tables 5 and 6, we find that multiple LLMs collectively outperform single LLMs in terms of Acc. Even the joint performance with the bottom-2 model is better than that of individual models, underscoring LLMs' diverse problem-solving capabilities. However, we note two limitations in joint modeling: (i) the joint performance with all LLMs may not always be the best (see *All LLMs* baseline Acc scores), as reported for the `MMLU` dataset, and (ii) joint modeling drastically increases inference latency costs (i.e., Lat), aligning with recent research (Li et al., 2024). In contrast, the proposed LLM routing aims to leverage joint LLM capabilities while minimizing latency by selecting the single best-suited LLM.

Can the upper bound performance of the classifier/clustering be equal to the Oracle model performance? This is possible in an ideal scenario where classifier/clustering routing algorithms are perfect and bias-free. However, in our case, the training data for the algorithms is small (∼9k in `GSM8K` and 15k in `MMLU`), which leads to suboptimal performance. Still, the multi-label classifier's upper bound (Acc) has achieved a higher score than any individual LLMs, which is also close to the Oracle model performance. We hypothesize that more training data for classification/clustering may bridge this gap.

Does router modeling with multi-label classifiers exhibit better performance than individual LLMs? Unfortunately, the proposed multi-label classifier with different confidence-based policies does not lead to better performance (i.e., Acc) than some individual LLMs. This may be due to the small training data for the classifier. However, it can be observed that the classifier's performance is better than most of the weak-performing LLMs and close to the top-performing LLM. This suggests that LLM routing is a promising direction that requires better classifier modeling.

What is the impact of different policies on LLM router modeling? We have proposed four policies based on the label confidence scores of the multi-label classifier. The best policy can push the model performance closer to the upper bound performance of the multi-label classifier. However, we observe that due to the imperfect classifier (which yields weighted F1 scores of 0.71 for GSM8K and 0.67 for MMLU), the predictions (and confidence scores) are skewed towards only a few labels (see Figure 4 in the Appendix) which leads to sub-optimal ACC score. The predictions-based policy is better than other policies; however, the classifier performance presents a serious bottleneck. We conclude that larger training data and the development of a better classifier are essential for improving the ACC scores. Small sizes of both GSM8K and MMLU datasets prevent further investigation of this question.

How does a separate classifier compare to a multi-label classifier for LLM routing? With relatively small and imbalanced training sets, separate classifiers for each LLM are more prone to over-fitting. Despite attempts to address this with measures like early stopping and weighted class-based loss, most individual models usually converge to the overall best performers such as `gemma-7b` on test split. Ultimately, with the argmax policy in place, the separate classifier-based routing model's performance is similar to that of the argmax policy of the multi-label classifier.

How does clustering-based LLM routing compare to other models? The cluster-level routing approach aims to select the best LLM for a group of similar query prompts. It assumes that the relative performance of LLMs for each cluster remains consistent between the training and test sets. We find that this assumption does not hold for many clusters (39 out of 50 for GSM8K and 28 out of 50 for MMLU). In general, the best-performing LLM for most clusters in the training set is the same as the best LLM overall. The impact of different feature extraction methods (TF-IDF vs. RoBERTa) was minimal, resulting in a similar performance to the MLC+ArgMax model.

What is the impact of LLM routing on inference latency? Table 3 provides the inference latencies for all LLMs, baselines, and LLM routing models in seconds per query, recorded using a single Nvidia A100 GPU. Ideally, the best routing policies should maximize model accuracy (while maintaining at least same-level latency) or minimize overall latency (with the best LLM accuracy maintained). For instance, the MLC+ArgMax latency is lower than the corresponding highest individual model latency (of `gemma-7b`) for GSM8K. However, as the routing classifiers overfit to the best LLMs on the training sets (`metamath-7b` for GSM8K and `gemma-7b` for MMLU), the overall latency, much like mean accuracy, differs very slightly from that of the best LLMs. These findings validate our claim that the proposed *LLM routing* model consistently maintains a latency score equal to or lower than any individual LLM.

Ablations with multi-label routing: In appendix Figure 5, we overview ablation tests for LLM routing using a multi-label classifier trained with best- and worst-performing LLMs across both datasets. Key insights include: (1) Increasing the number of top-performing LLMs improves oracle scores but has marginal effects on the classifier's upper bound or argmax policy. (2) Increasing the number of worse-performing LLMs results in higher scores across oracle, MLC's upper bound, and MLC+ArgMax policy model, highlighting the effectiveness of LLM routing.

5 Conclusions and Future Directions

This study investigates the feasibility of *LLM routing*, i.e., navigating input queries by efficiently selecting the most suitable single LLM from a pool of LLMs. Through extensive experimentation with multi-label and separate classifiers, as well as clustering across two challenging benchmarks, we conclude that (i) there are theoretical bounds that can be achieved with LLM routing that are much higher than individual models' performance, and (ii) routing LLMs is a feasible direction that works best with equally capable LLMs. However, if a few LLMs dominate, the router's performance degrades, even though it still outperforms weak LLMs. At the same time, the inference latency of the routing model is at least at the same level as that of single LLMs.

With these findings in mind, we envision future research to investigate the following directions: (1) collecting larger datasets for LLM routing design; (2) developing novel models for LLM routing to accommodate LLMs with diverse capabilities; (3) designing better routing policies with confidence scores; (4) incorporating LLM-specific features for improved modeling; and (5) scaling up using more diverse LLMs and benchmarks.

Limitations

One of the key limitations of the proposed routing model is the limited training data available for training different algorithms with varying policies, which can result in biased learning despite taking a number of precautionary measures. Another limitation is the extraction of answers from generated responses: despite utilizing our best answer extraction algorithm, we could only extract viable answers for 83% to 95% of queries (with different LLMs). For the remaining queries, the answers extracted with our algorithm may be invalid or incorrect. Next, the proposed model works well with equally capable LLMs but is not yet effective enough for LLMs that have very different capabilities.

Finally, although the inference latency of the proposed model is comparable to that of the most suitable single LLM, frequent switches between the LLMs (based on the input queries) necessitate loading most of the LLMs into memory, posing a limited memory issue. This issue is also observed with different emerging LLMs (Jiang et al., 2023, 2024) similarly to our case. At the same time, the problem of limited memory in the context of LLMs has been well studied (Alizadeh et al., 2023; Eliseev and Mazur, 2023), and the solutions developed are directly (or with minor adjustments) applicable to our modeling, thereby ensuring the practical usability of the proposed model. We leave investigation of such approaches to future work.

Ethics Statement

This paper introduces router modeling to effectively harness the power of LLMs with different capabilities. As the proposed routing models use LLMs, we must acknowledge that, independently of this research, there are certain risks that pertain to all LLMs, as such models may generate outputs that, although plausible, are factually incorrect or nonsensical. Such *hallucinations* can misguide decision-making and propagate biases, especially in critical scenarios where accuracy is vital. Without proper safeguards, widespread LLM adoption could worsen these concerns. Thus, it is essential to develop mechanisms to mitigate hallucination risks, ensuring responsible and beneficial deployment of these powerful models before adopting the proposed routing model.

Acknowledgments

We thank Jad Doughman and Ted Briscoe for insightful discussions about this research. We are grateful to the Campus Super Computing Center at MBZUAI for supporting this work. We also thank the anonymous reviewers for their valuable feedback.

References

Keivan Alizadeh, Iman Mirzadeh, Dmitry Belenko, Karen Khatamifard, Minsik Cho, Carlo C Del Mundo, Mohammad Rastegari, and Mehrdad Farajtabar. 2023. LLM in a flash: Efficient Large Language Model Inference with Limited Memory. *arXiv preprint arXiv:2312.11514*.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open LLM Leaderboard.

Christopher M Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv: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. *Advances in neural information processing systems*, 33:1877–1901.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2023. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ronan Collobert, Yoshua Bengio, and Samy Bengio. 2002. Scaling Large Learning Problems with Hard Parallel Mixtures. In *Pattern Recognition with Support Vector Machines*, pages 8–23, Berlin, Heidelberg. Springer Berlin Heidelberg.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing. In *The Twelfth International Conference on Learning Representations*.

David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2013. Learning Factored Representations in a Deep Mixture of Experts. *arXiv preprint arXiv:1312.4314*.

Artyom Eliseev and Denis Mazur. 2023. Fast inference of mixture-of-experts language models with offloading. *arXiv preprint arXiv:2312.17238*.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive Mixtures of Local Experts. *Neural Computation*, 3(1):79–87.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of Experts. *arXiv preprint arXiv:2401.04088*.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-Blender: Ensembling Large Language Models with Pairwise Ranking and Generative Fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178.

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.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for "mind" exploration of large scale language model society. *arXiv preprint arXiv:2303.17760*.

Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. 2024. More Agents Is All You Need. *arXiv preprint arXiv:2402.05120*.

Yixin Liu and Pengfei Liu. 2021. SimCLS: A Simple Framework for Contrastive Learning of Abstractive Summarization. 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 1065–1072.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP Models really able to Solve Simple Math Word Problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Ella Rabinovich, Samuel Ackerman, Orna Raz, Eitan Farchi, and Ateret Anaby-Tavor. 2023. Predicting Question-Answering Performance of Large Language Models through Semantic Consistency. *arXiv preprint arXiv:2311.01152*.

Sebastian Raschka. 2020. Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. *arXiv preprint arXiv:1811.12808*.

Mathieu Ravaut, Shafiq Joty, and Nancy Chen. 2022. SummaReranker: A Multi-Task Mixture-of-Experts Re-ranking Framework for Abstractive Summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4504–4524, Dublin, Ireland.

Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. 2017. Routing Networks: Adaptive Selection of Non-linear Functions for Multi-Task Learning. *arXiv preprint arXiv:1711.01239*.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2024. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *arXiv preprint arXiv:2402.07927*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2023. Quantifying Language Models' Sensitivity to Spurious Features in Prompt Design or: How I learned to start worrying about prompt formatting. *arXiv preprint arXiv:2310.11324*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. *arXiv preprint arXiv:1701.06538*.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. 2023. Large Language Model Routing with Benchmark Datasets. *arXiv preprint arXiv:2309.15789*.

KV Aditya Srivatsa and Ekaterina Kochmar. 2024. What makes math word problems challenging for llms? *arXiv preprint arXiv:2403.11369*.

Derek Tam, Anisha Mascarenhas, Shiyue Zhang, Sarah Kwan, Mohit Bansal, and Colin Raffel. 2023. Evaluating the Factual Consistency of Large Language Models Through News Summarization. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5220–5255, Toronto, Canada. Association for Computational Linguistics.

Hongyi Wang, Felipe Maia Polo, Yuekai Sun, Souvik Kundu, Eric Xing, and Mikhail Yurochkin. 2023. Fusing Models with Complementary Expertise. *arXiv preprint arXiv:2310.01542*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *Advances in neural information processing systems*, 35:22199–22213.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Dingjun Wu, Jing Zhang, and Xinmei Huang. 2023a. Chain of Thought Prompting Elicits Knowledge Augmentation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6519–6534, Toronto, Canada. Association for Computational Linguistics.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023b. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. *arXiv preprint arXiv:2308.08155*.

Wenxuan Zhang, Yue Deng, Bing Liu, Sinno Jialin Pan, and Lidong Bing. 2023. Sentiment Analysis in the Era of Large Language Models: A Reality Check. *arXiv preprint arXiv:2305.15005*.

A LLM Inference Latency

Prompt Type	LLM	GSM8K	MMLU
FCoT	`llama2-7b`	4.21	2.30
	`gemma-7b`	7.10	3.00
	`mistral-7b`	3.70	1.80
	`metamath-7b`	4.70	2.40
ZCoT	`gemma-7b-it`	0.70	1.00
	`llama2-13b-chat`	1.80	4.80
	`mistral-7b-it`	1.00	1.10

Table 4: Statistics on the inference latency (i.e., runtime in seconds) for various LLMs over 10 generations for each input query. The timings were recorded using a single Nvidia A100 GPU. `FCoT` denotes few-shot Chain-of-Thought, and `ZCoT` denotes zero-shot CoT. We have considered 5 examples for `FCoT` prompting.

B Prompting for LLM Sampling

The consideration of diverse LLMs and datasets contributed to the challenges in prompting, as there is no single uniform prompting approach across LLMs and datasets (Sclar et al., 2023). Considering recent findings about the appropriate usage of prompts (Sahoo et al., 2024) and those from our own experimentation, we have converged on the following prompting decisions:

- For non-chat LLMs, few-shot Chain-of-Thought (CoT; Wei et al. (2022)) prompting works better than zero-shot (Kojima et al., 2022) for both datasets. We used 5 few-shot examples. The few-shot prompting leads to over 95% *viable* answers (except for `llama2-7b` LLM, which has the viability score of 83%) in generated solutions. A *viable* answer is a single numeric/alphabetic answer that can be extracted from the generated solution using extraction algorithms (see Appendix C) to compare with the reference answer. The viable answer can be correct or incorrect.

- For chat LLMs, few-shot CoT distracts the generation, which leads to unexpected outputs. The zero-shot CoT works best. We utilize different models' chat-templates from Hugging Face[6] to ensure correctness. The viability of answer extraction for chat models is 92%.

The sample zero-shot and few-shot CoT prompt templates are presented in Figure 2.

[6] https://huggingface.co/docs/transformers/en/chat_templating

C Answer Extraction from LLM Responses

The adapted prompting approaches used in our LLM queries are designed to instruct LLMs to specify their final answers at the very end of each of their responses. We thus use a simple answer extraction policy of selecting the last mentioned numerical value (for GSM8K) and multiple-choice option (for MMLU) from the generated responses. Responses failing to report any final answer are regarded as invalid and counted as incorrect answers. For MMLU, we evaluate the extracted options directly against the annotated correct options (among 'A', 'B', 'C', and 'D') in the dataset. For GSM8K, questions where the absolute difference between the ground truth and predicted numerical answers are less than $\epsilon = 0.1$ are considered to be solved correctly. This threshold was set to accommodate instances where model-generated real-valued answers differ slightly from the expected answer.

Lessons Learned: It is observed that sometimes the expected answer is present in one of the last sentences of the response instead of at the very end. We extracted all such answers as well. Allowing a 0.1 absolute error difference leads to more accurate answers.

D Implementation, Hyperparameters, and Hardware Details

Querying LLMs We use the vLLM package[7] to query LLMs. All models were queried with a temperature of 0.8 and a max token length of 2000. Each question prompt was queried 10 times with different initialization seeds. We used a single Nvidia A100 GPU for all runs. Querying each dataset once took approximately 1-2 hours.

Training Routing Classifiers We use the HuggingFace library[8] for loading and tuning all pretrained Transformer encoders in our experiments. Each model was trained for 10 epochs, with an initial learning rate of 2e-5, warmup ratio of 0.1, and class-balanced CrossEntropy loss. The training checkpoint with the lowest validation loss was selected for inference.

E Detailed Results for Routing Models

See Figures 3-5 and Tables 5-6.

[7] https://github.com/vllm-project/vllm
[8] https://huggingface.co/

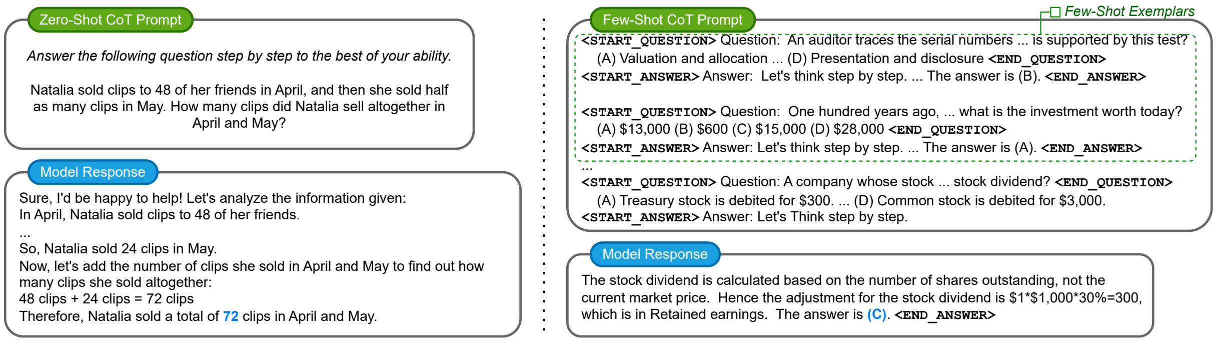
Models	ACC (%)	LAT (sec)
Oracle	87.18	3.46
Random	55.37	3.52
gemma-7b	71.11	7.10
metamath-7b	67.55	4.70
mistral-7b	59.74	3.70
mistral-7b-it	50.41	1.00
llama2-13b-chat	46.70	1.80
gemma-7b-it	36.84	0.70
top-2 LLMs	81.80	11.80
top-3 LLMs	84.00	15.5
top-4 LLMs	85.82	16.5
top-5 LLMs	86.03	18.3
bottom-2 LLMs	55.64	2.50
bottom-3 LLMs	67.02	3.50
bottom-4 LLMs	75.51	7.20
bottom-5 LLMs	79.91	11.90
All LLMs	74.37	19.00
Upper Bound of MLC	79.68	5.16
MLC + Argmax policy	67.62	4.76
MLC + Random policy	67.47	4.76
MLC + Prediction policy	67.70	4.77
MLC + Sorted Pred policy	59.90	4.77
SC + Argmax policy	67.55	4.70
Clustering + TF-IDF	67.55	4.70
Clustering + RoBERTa	67.55	4.70

Table 5: Performance of different routing models on GSM8K data. ACC: mean accuracy with MAJ@10 (%), LAT: LLM inference latency in seconds per query (10 generations for each query), MLC: multi-label classifier, SC: separate classifiers, and top-k: best k performing models. All other notation is the same as for Table 3.

F Related Work

Model Diversity Several surveys (Bommasani et al., 2021; Minaee et al., 2024, inter alia) suggest that LLMs can develop emergent capabilities. Specifically, this suggests that models can show behavior and demonstrate skills beyond explicitly constructed ones. By virtue of differing training data, models may exhibit a wide variety of domain expertise. Jiang et al. (2023) demonstrates that no single open-source LLM outperforms other models across popular benchmarks. This further motivates the need to develop ensembling or routing methods aimed at improving the combined performance of a pool of LLMs with a diverse range of abilities.

Model Selection A fundamental step in routing queries within an ensemble of models is to estimate the extent of overlap between the capabilities of the LLMs in the candidate pool with those deemed necessary to resolve an input query. Model selection in the context of LLM routing greatly differs from its traditional form in ML (Bishop, 2006; Raschka, 2020), wherein the training and test datasets are similar in distribution. Training data



Zero-Shot CoT prompt for GSM8K

Few-Shot CoT prompt for MMLU

Figure 2: Sample zero-shot Chain-of-Thought (CoT) prompt template for a chat (or instruction-tuned) LLM and few-shot Chain-of-Thought (CoT) prompt template for a standard LLM.

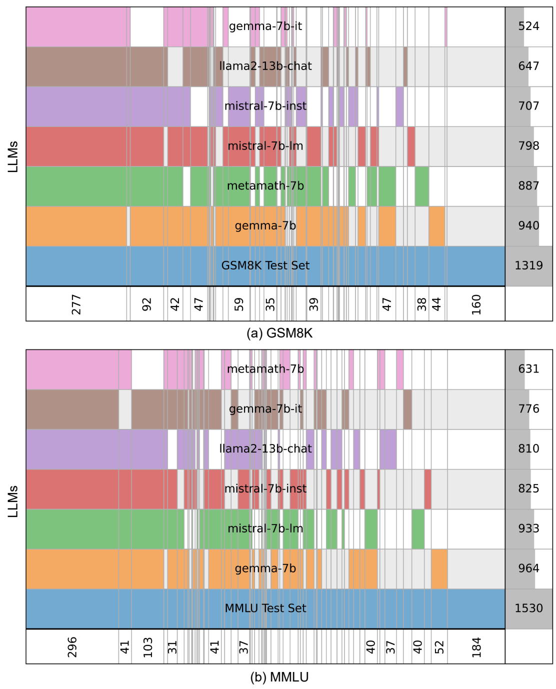


Figure 3: Distribution of queries from the GSM8K and MMLU test sets solved (score 1 with MAJ@10) by each LLM. The counts at the bottom of each figure denote the number of questions in each chunk, and those on the right denote the total number of questions solved by each LLM.

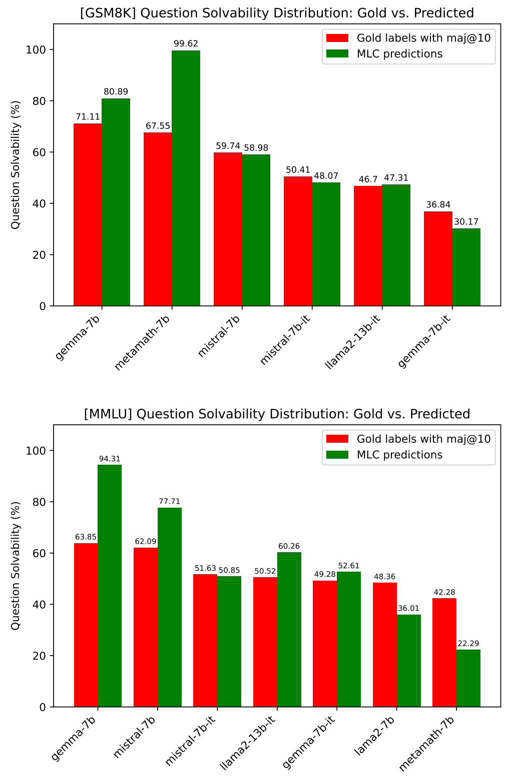


Figure 4: LLMs "solvability" distribution. The gold label scores are obtained with MAJ@10, and prediction label scores are obtained with a multi-label classifier.

for LLMs include massive corpora spanning trillions of tokens with relatively straightforward learning objectives like next-token prediction (Brown et al., 2020). Test data, on the other hand, often involves highly structured tasks like reasoning and question answering (Hendrycks et al., 2021; Cobbe et al., 2021; Joshi et al., 2017), summarization (Tam et al., 2023), and classification (Zhang et al., 2023), which may not be very prevalent in corresponding training data. This makes gauging the pain points of resolving a complex query non-trivial. Furthermore, studies like Rabinovich et al. (2023) and Srivatsa and Kochmar (2024) suggest that certain aspects of the prompt phrasing, i.e., its length and readability, significantly impact LLMs' ability to tackle the underlying tasks.

LLM Ensembling Previous attempts at ensembling and routing of LLMs aim to tackle one of two tasks: **(1)** Opting between LLM generations to select the best response. Liu and Liu (2021); Ravaut

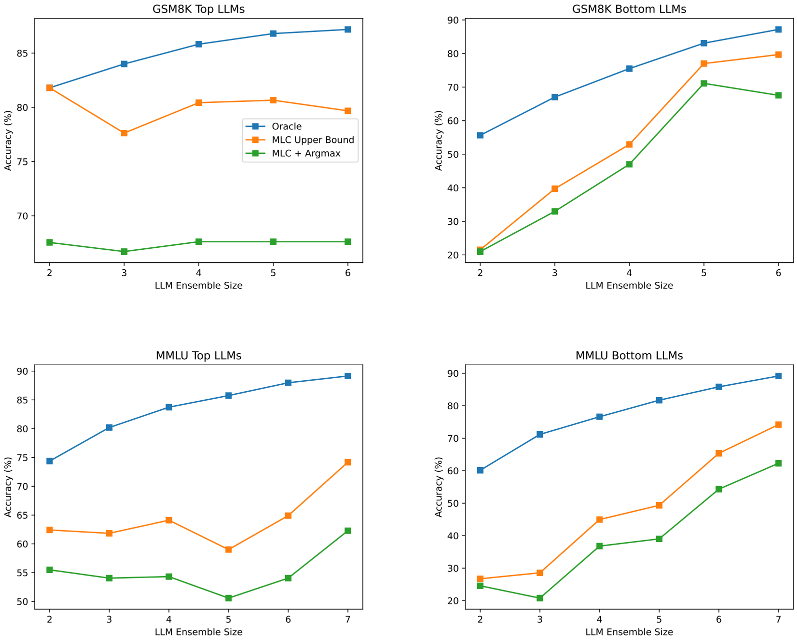


Figure 5: Different ablation configurations for LLMs for GSM8K and MMLU datasets.

Models	Acc (%)	Lat (sec)
Oracle	89.15	1.89
Random	52.50	2.35
gemma-7b	63.85	3.00
mistral-7b	62.09	1.80
mistral-7b-it	51.63	1.10
llama2-13b-chat	50.52	4.80
gemma-7b-it	49.28	1.00
llama2-7b	48.36	2.30
metamath-7b	42.28	2.40
top-2 LLMs	73.47	4.80
top-3 LLMs	79.54	5.90
top-4 LLMs	83.72	10.70
top-5 LLMs	85.75	11.70
top-6 LLMs	87.88	14.0
bottom-2 LLMs	60.13	4.70
bottom-3 LLMs	71.17	5.70
bottom-4 LLMs	78.10	10.50
bottom-5 LLMs	81.69	11.60
bottom-6 LLMs	83.11	13.40
All LLMs	60.39	16.40
Upper Bound of MLC	77.18	1.94
MLC + Argmax policy	62.28	2.95
MLC + Random policy	58.16	2.86
MLC + Prediction policy	63.85	2.95
MLC + Sorted Pred policy	48.36	2.92
SC + Argmax policy	62.87	2.94
Clustering + TF-IDF	61.76	2.83
Clustering + RoBERTa	61.76	2.83

Table 6: Performance of different routing models on the MMLU data. Acc: mean accuracy with MAJ@10 (%), Lat: LLM inference latency in seconds per query (10 generations for each query), MLC: multi-label classifier, SC: separate classifiers, and top-k: best k performing models. All other notation is the same as for Table 3.

et al. (2022); Jiang et al. (2023) train models to rank or classify the most suitable response for a given query. However, this requires querying all LLMs in the model pool for each query during inference time. This can become computationally expensive with a large number of LLMs in the candidate pool. **(2)** Building routing networks (Rosenbaum et al., 2017) that utilize only a subset of parameters of a model or a subset of experts from a pool of candidate models. For example, Jiang et al. (2024) employ a Mixture-of-Experts (MoE) (Jacobs et al., 1991; Collobert et al., 2002; Eigen et al., 2013) model with 8 experts, wherein only 2 experts are accessed at each model layer to produce the next token. This, however, requires pre-training the model weights, which incurs large computing and data costs. Alternatively, HybridLLM (Ding et al., 2024), Shazeer et al. (2017), and Shnitzer et al. (2023) train separate classifiers which select the best LLM(s) for each input query.

This paper aims to create and study a sparse routing network for selecting the best LLM from a pool of more than two LLMs for each example. The routing network only needs to tune an extra Transformer-based classifier without needing to pre-train or fine-tune the LLMs. Furthermore, we also incorporate the former task by measuring the response quality (through accuracy) and determining if it can outperform the individual experts (LLMs) in the pool.

The Paradox of Preference: A Study on LLM Alignment Algorithms and Data Acquisition Methods

Rishikesh Devanathan*, **Varun Nathan** and **Ayush Kumar**
{rishikesh.devanathan, varun.nathan, ayush}@observe.ai
Observe.AI
Bangalore, India

Abstract

This research investigates the impact of preference annotation acquisition methods on the performance of LLM alignment algorithms, including Direct Preference Optimization (DPO), Identity Preference Optimization (IPO), and Conservative DPO (cDPO), compared to Supervised Fine-Tuning (SFT) in NLP tasks. We analyze the influence of LLM and human-based preferences on algorithm performance, considering data volume and quality. Additionally, we assess DPO's vulnerability to overfitting and IPO's resilience against it, addressing four main research questions. Using the GAIR dataset and Zephyr-7b as the SFT model, we reveal unexpected negative outcomes. Specifically, DPO trained on LLM preferences outperforms human preferences, contrary to expectations. Moreover, there's no correlation between preference data volume or quality and algorithm performance. Contrary to expectations, DPO shows no overfitting in both human and LLM preference datasets. Surprisingly, cDPO doesn't fare better than DPO under flip noise. Our findings highlight the complexities of preference annotation methods and underscore the importance of scrutinizing negative results in NLP algorithm research.

1 Introduction

Large language models (LLMs) have proven their capacity to amass broad knowledge by simply maximizing the likelihood of human-written text but this objective isn't sufficient to generate responses that are safe, helpful and aligned with human preferences. Methods based on Reinforcement Learning with Human Feedback (RLHF), including Proximal Policy Optimization (PPO) (Schulman et al., 2017), aim to align LLMs with human preferences, a theme also explored in other papers (Ouyang et al., 2022; Askell et al., 2021; Bai et al., 2022a; Touvron et al., 2023). Direct Preference Optimization (DPO) (Rafailov et al., 2023) was later shown to train policies in a single stage, treating it as a classification task using human preference data. It's favored over PPO for its ability to handle reward translation issues well and consistently achieve high rewards across different levels of KL divergence in generated text.

Due to the expensive nature of collecting human annotations, LLM preferences serve as a substitute for human preferences in generating synthetic datasets (Chiang and Lee, 2023). Reinforcement Learning from AI Feedback (RLAIF) (Bai et al., 2022b) provides a promising alternative by leveraging a powerful off-the-shelf LLM to generate preferences for large-scale model training. The use of LLM preferences in dataset creation (Lee et al., 2023) has shown comparable performance between RLAIF and RLHF across various tasks, with performance degradation attributed to dataset quality issues, as evidenced by human-agreement scores. Conservative DPO (cDPO) [1] addresses these challenges by adopting a conservative target distribution, minimizing error probability, and deriving a loss function to ensure alignment between model preferences and observed preferences. The scarcity of diverse preference datasets poses a challenge for RLHF and feedback learning research. ULTRAFEEDBACK (Cui et al., 2023) addresses this challenge by providing an extensive, high-quality, and diversified preference dataset.

While widely adopted in preference optimization, DPO is susceptible to overfitting as observed by Tunstall et al. (2023) in the initial epoch of Zephyr-7B DPO training, but noted improved performance with further epochs. The IPO paper (Azar et al., 2023) discovered that RLHF and DPO are prone to overfitting due to relying on the assumption that pairwise preferences can replace ELo-

* Work done during internship at Observe.AI

[1] https://ericmitchell.ai/cdpo.pdf

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 135–147
June 20, 2024 ©2024 Association for Computational Linguistics

scores through Bradley-Terry modeling. To mitigate this, IPO introduces a regularizing term controlling log-likelihood ratios to address overfitting to the preference dataset.

A literature gap exists in exploring how training data volume influences LLM alignment algorithms, DPO and IPO. Empirical evidence is lacking on IPO's ability to counter DPO's overfitting, and studies on data quality's impact on DPO, and cDPO's effectiveness in addressing it, are scarce. It's essential to investigate the influence of preference annotation methods (LLM vs. human preferences) on these factors and the performance of LLM alignment algorithms, including DPO, cDPO, and IPO, given the increasing use of LLM preferences.

In this work, we investigate how the method of preference annotation acquisition affects the critical performance factors influencing the effectiveness of LLM alignment algorithms and seek to address the following research questions:

- **RQ1:** How does the choice of preference annotation acquisition method influence the performance of DPO and IPO in comparison to SFT?
- **RQ2:** What is the effect of data volume on the performance of DPO and IPO? Does the relationship depend on the preference annotation acquisition method?
- **RQ3:** What is the effect of data quality on the performance of DPO and cDPO? Does the relationship depend on the preference annotation acquisition method?
- **RQ4:** To what extent does DPO suffer from overfitting, and can IPO withstand it? How does the preference annotation acquisition method impact this phenomenon concerning both loss functions?

We demonstrate unexpected superiority of LLM trained with DPO on LLM preferences over human preferences. Performance shows no correlation with data volume or quality. DPO doesn't exhibit overfitting issues, while cDPO doesn't improve under noise. Our findings highlight challenges in preference annotation and aligning LLMs.

2 Implementation Details

We choose Zephyr-7B as our SFT model and GAIR (Li et al., 2024) as our preference dataset, containing both human and LLM preferences, for our experiments. MT Bench (Zheng et al., 2023) is used to evaluate our models while GPT4-Turbo is chosen as the LLM for obtaining synthetic preferences. Further details on our choices are provided in Section A.1, including hyperparameter specifics.

3 Results and Analysis

In this section, we provide a comprehensive analysis of the performance evaluation results, shedding light on the key observations made during our study.

3.1 RQ1 (Preference model performance)

To investigate this, we independently fine-tuned Zephyr-7B (SFT) using preferences from both GPT4-Turbo and humans in the GAIR dataset. In Table 1, the IPO model trained on human preferences, as anticipated, outperforms its GPT4-Turbo-trained counterpart according to the MT Bench score. However, contrary to expectations, the DPO model trained on GPT4-Turbo preferences outperforms its human-trained counterpart according to the MT Bench score. We speculate that GPT4, acting as the MT Bench judge, might show bias towards responses from GPT4-Turbo-trained models. To verify, we collect predictions from both models on MT Bench, comprising 160 samples, and shared them with our in-house annotation team of three members. Model names were concealed, and annotators chose from options 'model 1' (GPT4-Turbo preference trained model), 'model 2' (human preference trained model), or 'equal' based on the quality of the generated output. We opt for a majority vote to determine the final preference, and the inter-annotator agreement score, calculated using Fleiss' Kappa (Fleiss et al., 1971), was measured at 0.64. As shown in Table 2, the model trained on GPT4-Turbo preferences was preferred in 63 of 160 samples with a much higher win rate of 39.4%, suggesting alignment between MT Bench scores and human annotation. This negative outcome of model trained on LLM preference data outperforming the one trained on human preference data prompts a crucial inquiry regarding the superiority of human preferences over those sourced from LLMs and the necessary measures to guarantee the quality standards of human-collected data.

3.2 RQ2 (Effect of data volume)

To investigate this, we independently fine-tune Zephyr-7B (SFT) using DPO and IPO losses on

Algorithm	GPT4-Turbo	Human
SFT (Baseline)	6.753	6.753
Preference Model (DPO)	6.994	6.722
Preference Model (IPO)	5.125	5.484

Table 1: Benchmarking performance of DPO and IPO by Preference annotation acquisition method using MT Bench scores

Model	# Wins	# Ties	Win Rate
DPO (GPT4-Turbo)	63	51	39.38%
DPO (Human)	46	51	28.75%

Table 2: Results from human annotation of DPO model trained on GPT4-Turbo and Human preferences

Data Volume (% Train Data)	GPT4-Turbo		Human	
	Loss = DPO	Loss = IPO	Loss = DPO	Loss = IPO
100%	6.994	5.125	6.722	5.484
75%	6.756	5.741	6.544	6.300
50%	6.878	6.766	6.897	6.692
25%	6.788	6.953	6.819	6.928

Table 3: Benchmarking performance of DPO and IPO models by preference annotation acquisition method when trained on different data volumes using MT Bench scores

sampled datasets with varying proportions of preferences from both GPT4-Turbo and humans in the GAIR dataset. Contrary to the anticipation of improved generalization with increased data diversity, this pattern is absent in DPO and IPO models trained on both types of preferences (Table 3). Neither GPT4-Turbo-Preference-trained nor human-preference-trained DPO and IPO models demonstrate a monotonic relationship with data volume, suggesting that augmenting preference data volume may not necessarily enhance model performance. Notably, DPO and IPO models trained on 25% human preference data outperform those trained on the entire dataset, hinting at potential overfitting issues. We conduct an exhaustive examination into the susceptibility of DPO to overfitting, with detailed results emphasized in 3.4.

Table 3 also demonstrates that models trained with IPO underperform those trained with DPO across sample proportions of 100%, 75%, and 50% in both GPT4-Turbo and human preference datasets. Upon conducting a hyperparameter sweep over a fine-grained range for the DPO and IPO models trained on 100% of the human preference dataset, significant uplift in performance was observed for both IPO and DPO models post-tuning β as indicated in Table 8. However, we see that DPO still outperforms IPO, indicating the inefficacy of IPO in surpassing DPO despite extensive tuning of β. Due to the high cost of running these experiments and the limited effectiveness of IPO, we did not extend the tuning exercise to other configurations.

This finding highlights the valuable insight for ML researchers and scientists in enterprises using DPO for preference modeling. It also underscores the challenges involved in exploring alternative loss functions such as IPO to enhance performance with limited preference data.

3.3 RQ3 (Effect of data quality)

To tackle this issue, we independently fine-tune Zephyr-7B (SFT) using DPO and cDPO losses on sampled datasets with varying levels of flip noise introduced into preferences from both GPT4-Turbo and humans in the GAIR dataset. Flip noise is introduced by swapping the chosen response and the rejected response for a selected percentage of prompts. Despite the anticipation that models would exhibit better generalization with higher data quality, this pattern is not evident in DPO and cDPO models trained on both types of preferences (Table 4). Intriguingly, DPO models trained with 25% flip noise outperform those trained on clean data across GPT4-Turbo and human preferences, while the cDPO model only marginally outperforms it when trained on 50% flip noise data.

Moreover, Table 4 indicates that models trained with cDPO consistently exhibit inferior performance compared to those trained with DPO across all configurations and datasets. This contradicts expectations set by the cDPO paper, which suggests that cDPO's ability to optimize to a fixed delta from the reference model and then halt likely enhances its stability compared to the original DPO loss, making it more effective when dealing with noisy

Data Quality (% Flip Noise)	GPT4-Turbo		Human	
	Loss = DPO	**Loss = cDPO**	**Loss = DPO**	**Loss = cDPO**
0%	6.994	6.994	6.722	6.722
5%	6.956	6.733	6.759	6.559
25%	7.013	6.313	6.731	6.284
50%	7.081	6.372	6.703	6.344
75%	6.984	5.456	6.584	5.378

Table 4: Benchmarking performance of DPO and cDPO models by preference annotation acquisition method when trained on datasets with different flip noise ratios using MT Bench scores

# Steps	GPT4-Turbo				Human							
	Loss = DPO		**Loss = IPO**		**Loss = DPO**		**Loss = DPO (Tuned)**		**Loss = IPO**		**Loss = IPO (Tuned)**	
	Training Loss	MT Bench Score	Training Loss	MT Bench Score	Training Loss	MT Bench Score	Training Loss	MT Bench Score	Training Loss	MT Bench Score	Training Loss	MT Bench Score
1/4	0.212	6.809	11.796	6.578	0.283	6.638	0.539	6.969	14.775	6.669	0.577	6.919
2/4	0.037	6.859	5.959	5.244	0.057	6.744	0.505	6.919	9.322	4.677	0.368	7.056
3/4	0.024	6.813	4.056	4.781	0.038	6.728	0.253	6.981	6.648	5.874	0.274	6.953
4/4	0.018	6.994	3.231	5.125	0.031	6.722	0.219	7.184	5.415	5.484	0.241	7.113

Table 5: Impact of Overfitting on DPO and IPO Models at 100% Data Volume by Preference Annotation Acquisition Method

training data. Upon conducting a thorough hyperparameter sweep over a finely grained range for both DPO and cDPO models trained on the human preference dataset with 5% flip noise, significant performance enhancements were observed for both after β tuning as indicated in Table 9. However, DPO continues to surpass cDPO, indicating the limited efficacy of cDPO even after extensive β tuning. Due to the substantial expenses involved in running these experiments and the limited effectiveness of cDPO, we discontinued extending the tuning process to other configurations.

This negative outcome holds considerable significance for researchers and professionals in organizations utilizing DPO for preference modeling in noisy datasets.

3.4 RQ4 (DPO and IPO overfitting)

Our objective is to validate the hypothesis that DPO is susceptibile to overfitting and IPO is resilient against it (Azar et al., 2023). We conduct empirical validation by independently fine-tuning Zephyr-7B (SFT) using DPO and IPO losses on 100% of preferences from both GPT4-Turbo and humans in the GAIR dataset. Overfitting is assessed by monitoring training loss and evaluation scores of checkpoints at intervals of 25% of the training steps on MT Bench.

Table 5 reveals that DPO exhibits overfitting only when trained on human preferences with the default β of 0.1, contrasting IPO, which exhibits overfitting when trained on both types of preferences. As suggested in the paper, we hypothesised that tuning beta would help mitigate overfitting in IPO trained model. As expected, when examining models trained with a tuned β, a different pattern emerges, where both DPO and IPO models trained on human preference data do not display overfitting. Thus, the key negative result we observe is that tuning β (0.00625) helps mitigate overfitting in DPO when trained on human preferences, providing valuable insights for ML researchers and industry practitioners employing DPO and IPO for preference modeling with limited data.

4 Conclusion

We analyze the influence of data quantity on DPO and IPO, utilizing LLM preferences and human preferences. Surprisingly, there's no linear correlation between data quantity and performance. Similarly, the impact of data quality on DPO and cDPO, using both LLM preferences and human preferences, also lacks a linear trend with performance. Contrary to expectations, DPO trained on LLM preferences outperforms its human-trained counterpart. Additionally, IPO fails to outperform DPO across various data volumes, while cDPO struggles to address induced flip noise in preferences. Interestingly, DPO shows no signs of overfitting when trained on both LLM and human preference datasets. These findings prompt further research to enhance the resilience and effectiveness of LLM alignment algorithms in preference modeling.

5 Limitations

This study offers significant insights into the performance of LLM alignment algorithms and the influence of preference annotation acquisition methods, but it is not without its limitations. First, the research is grounded in a specific set of LLM alignment algorithms, namely DPO, IPO, and cDPO. The results may not extend to other alignment algorithms like KTO (Ethayarajh et al., 2024). Future studies could broaden the scope by examining the performance of different algorithms for a more holistic understanding of the field. Second, the GAIR training dataset and MT Bench evaluation dataset were used in this study. The outcomes might vary with the use of different datasets, hence, extrapolating these findings to other contexts should be done with caution. Third, the Zephyr-7b, a decoder-only model, was used as the underlying SFT model, and GPT4-Turbo was used as the source in GAIR for acquiring LLM-based preferences. The outcomes might differ with the use of other models. Specifically, the trends observed may not necessarily apply to other SFT models within the same architectural class or different architectural classes such as encoder-decoder models. Fourth, the study did not find a correlation between the volume or quality of preference data and algorithm performance. However, this does not exclude the possibility of other factors influencing algorithm performance. Additional research is required to identify these potential factors. Fifth, the study found that DPO trained on LLM preferences outperforms human preferences, which was unexpected. This raises questions about the validity of human preferences as a performance benchmark for algorithms. Future research should delve deeper into this issue. Lastly, the study found no evidence of overfitting in DPO when trained on both LLM and human preference datasets. However, this finding should be interpreted cautiously as overfitting is a multifaceted issue influenced by various factors, including model complexity, training dataset size, and data noise. Further research is needed to fully comprehend the conditions that may lead to overfitting.

In conclusion, while this study offers valuable insights into the performance of LLM alignment algorithms and the impact of preference annotation acquisition methods, these findings should be considered in light of the aforementioned limitations. Future research should strive to address these limitations for a more comprehensive understanding of the field.

References

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Benjamin Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. 2021. A general language assistant as a laboratory for alignment. *CoRR*, abs/2112.00861.

Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. A general theoretical paradigm to understand learning from human preferences. *CoRR*, abs/2310.12036.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, Benjamin Mann, and Jared Kaplan. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *CoRR*, abs/2204.05862.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosiute, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemí Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022b. Constitutional AI: harmlessness from AI feedback. *CoRR*, abs/2212.08073.

David Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 15607–15631. Association for Computational Linguistics.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and

Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback. *CoRR*, abs/2310.01377.

Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. 2022. Understanding dataset difficulty with $\mathcal{V}$-usable information. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 5988–6008. PMLR.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. KTO: model alignment as prospect theoretic optimization. *CoRR*, abs/2402.01306.

J.L. Fleiss et al. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. 2023. RLAIF: scaling reinforcement learning from human feedback with AI feedback. *CoRR*, abs/2309.00267.

Junlong Li, Fan Zhou, Shichao Sun, Yikai Zhang, Hai Zhao, and Pengfei Liu. 2024. Dissecting human and LLM preferences. *CoRR*, abs/2402.11296.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of LM alignment. *CoRR*, abs/2310.16944.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: less is more for alignment. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

A Appendix

A.1 Implementation Details for LLM Alignment Experiments

In this section, we elaborate on the implementation details of our study, exploring how variations in the quality and quantity of preference data impact the performance of DPO, IPO, and cDPO, alongside the influence of preference annotation acquisition methods.

We opt for Zephyr-7B, a decoder-only model based on Mistral-7B, as our SFT model due to its top-ranking performance in MT Bench (top 5 in the list of models with a non-proprietary license) and accessibility in the HuggingFace model repository under the apache-2.0 license.

We employ the GAIR preference dataset (Li et al., 2024), comprising 5.24K curated conversations with pairwise human preferences from 13K unique IP addresses on the Chatbot Arena, collected between April and June 2023. Additionally, binary preference labels are gathered from 32 LLMs, incorporating 2 proprietary and 30 open-source models. With 29 defined properties, each response is annotated using Likert scale ratings or property-specific annotations. This dataset is selected primarily for its inclusion of both human and LLM preferences. Furthermore, the renowned LIMA paper (Zhou et al., 2023) originates from the same organization that released this dataset.

We also conduct experiments on two additional preference datasets: Ultrafeedback (Cui et al., 2023), comprising 61K prompts with preferences sourced from GPT4, and the Stanford Human Preferences Dataset (SHP) (Ethayarajh et al., 2022), extracted from posts and user comments across 18 subreddits containing human preferences, totaling 349K samples, which we downsample to 100K samples by filtering for those with a score ratio greater than 2 for experimentation. These datasets are selected for their extensive scale and diversity compared to other datasets.

Models are evaluated using MT Bench (Zheng et al., 2023), a curated benchmark featuring 80 high-quality, multi-turn questions designed to evaluate conversation flow and instruction-following capabilities in multi-turn dialogues. GPT-4 rates MT Bench outputs on a scale of 1-10, with higher scores indicating better performance. Refer to Table 15 for the domains considered in the datasets. Average MT Bench scores across questions and turns are reported for all experiments.

We fine-tune all alignment models for two to three epochs, following the approach in Tunstall et al. (2023). Adam optimizer with betas of (0.9, 0.999) and epsilon of 1e-08 is utilized. A linear learning rate scheduler with a peak rate of 5e-7 and 10% warmup steps is applied. Models are trained with a global batch size of 16, using β = 0.1 to control deviation from the reference model. A hyper-parameter sweep for β is performed over the range $\in \{1e-3, 2.5e-3, 5e-3, 6.25e-3, 1e-2, 5e-2, 1e-1, 1.5e-1, 2e-1, 5e-1, 9e-1\}$ for four settings: training DPO / IPO models on 100% data volume + 0% flip noise and DPO / cDPO models on 100% data volume + 5% flip noise. β tuning is specifically focused on due to its significant impact on model performance. Given the high training cost, β tuning is not conducted for all experiments. Experiments are conducted on an AWS p4de.24xlarge instance with eight GPUs, each with 80 GB of memory. A single training run takes 3-4 hours on average, costing approximately $140-190. Results are reported as the mean of 4 runs.

Dataset: `https://huggingface.co/datasets/GAIR/preference-dissection`

Training Code: `https://github.com/huggingface/alignment-handbook/tree/main`

Evaluation Code: `https://github.com/lm-sys/FastChat`

A.2 Error Analysis

In our study, we encountered several unexpected results that contradicted our initial hypotheses. This section provides an in-depth error analysis to understand these observations and their potential causes.

Firstly, we posited that DPO performance would be superior when trained on human preferences compared to LLM preferences. However, our findings contradicted this hypothesis. One plausible explanation for this unexpected outcome could be the inherent biases present in human preferences, which may not align with the objective function of the DPO algorithm. Moreover, there may be inherent limitations in the methodology used to collect human annotations.

An example of this discrepancy is evident in the performance of the DPO model trained on GPT4-Turbo preferences versus human preferences, particularly in the task of coding, as illustrated in figures 1 and 2. It is conceivable that the expertise levels of the human annotators selected for this task were not carefully considered.

Additionally, our analysis revealed instances of hallucinations (Row 3 in Table 16) and the generation of incomplete or redundant responses for Math questions (Row 6 in Table 16) by the DPO model trained on human preferences. These discrepancies may be attributed to various biases inherent in human preferences or inconsistencies in annotation practices.

Conversely, LLM preferences may exhibit greater consistency or comprehensiveness, thereby yielding superior performance. Further investigation is warranted to elucidate this discrepancy.

Secondly, we observed no discernible correlation between the volume or quality of preference data and the performance of the alignment algorithms. This finding challenges the widely held assumption that larger, higher-quality datasets invariably lead to improved performance. One potential contributing factor to this discrepancy could be the possible absence of independence and identical distribution (iid) in the data sourced from GAIR (Li et al., 2024), which may have influenced the outcomes of our experiments.

As depicted in figures 3 and 4, the disparities in performance among models trained on varying data volumes or with different proportions of flip noise are not uniformly distributed across the domains in MT Bench (Zheng et al., 2023). To delve deeper into this phenomenon, we manually mapped the domains in GAIR (Li et al., 2024) and MT Bench (Zheng et al., 2023), as illustrated in Table 6. Subsequently, we aggregated the data volume from GAIR (Li et al., 2024) according to the distinct domains in MT Bench (Zheng et al., 2023), as presented in Table 7.

As demonstrated in Table 7, there exists an imbalance in the distribution of samples within GAIR (Li et al., 2024) across the various domains in MT Bench (Zheng et al., 2023). This non-uniform distribution could potentially skew the results of our experiments on data quantity and quality. Additionally, it is plausible that there are diminishing returns once a certain threshold of data volume is surpassed.

Thirdly, contrary to our initial expectations, the DPO algorithm did not exhibit indications of overfitting on either the human or LLM preference datasets. This suggests the possibility that our methods for detecting overfitting may not have been sufficiently sensitive. Moreover, the relatively small volume of the GAIR (Li et al., 2024) dataset, consisting of approximately 5.2K samples, may have biased the results pertaining to DPO overfitting. It is conceivable that the dataset lacked the requisite data volume to effectively capture the onset of overfitting.

Furthermore, the decision to train for only 2-3 epochs might have been too brief to provoke overfitting, particularly because the learning rate was appropriately set. We opted for this duration based on the findings of Tunstall et al. (2023), who reported observing overfitting after a single epoch. However, the divergence in observed behaviors could be attributed to the differences in the nature and size of the datasets.

It is worth noting that models often necessitate additional training iterations before overfitting manifests, as they gradually adapt not only to the underlying pattern but also to the noise present in the training data. Consequently, further investigation is warranted to ascertain the precise underlying cause of these observations.

Lastly, cDPO did not perform better than DPO under flip noise conditions. This was surprising as cDPO is designed to be more conservative and thus more resilient to noise. One possible explanation could be that the flip noise in our dataset was not significant enough to differentiate the performance of DPO and cDPO. Alternatively, there might be other types of noise or errors that cDPO is not equipped to handle.

In conclusion, our error analysis has revealed several unexpected findings that challenge common assumptions in LLM alignment algorithm research. These findings underscore the importance of rigorous error analysis and the need for further research to understand the complexities of preference annotation methods.

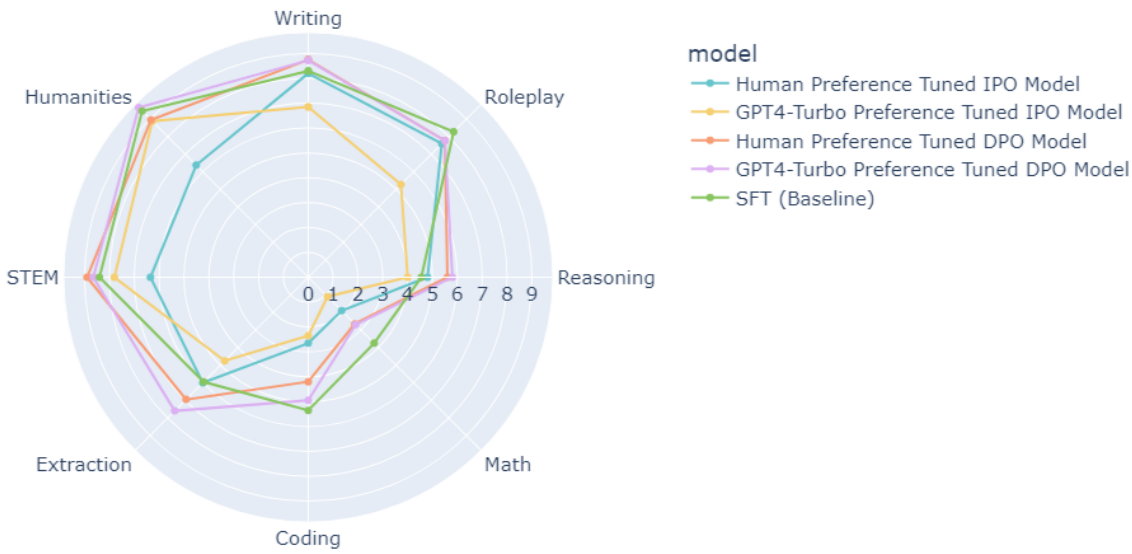


Figure 1: Analysis of GPT4 Ratings by Domains in MT Bench: DPO and IPO Models trained on GPT4-Turbo vs Human Preferences. Notably, the DPO model trained on GPT4-Turbo preferences excels over its counterpart trained on Human preferences in domains such as Coding, Extraction, Reasoning, and Humanities, while demonstrating competitive performance in other areas.

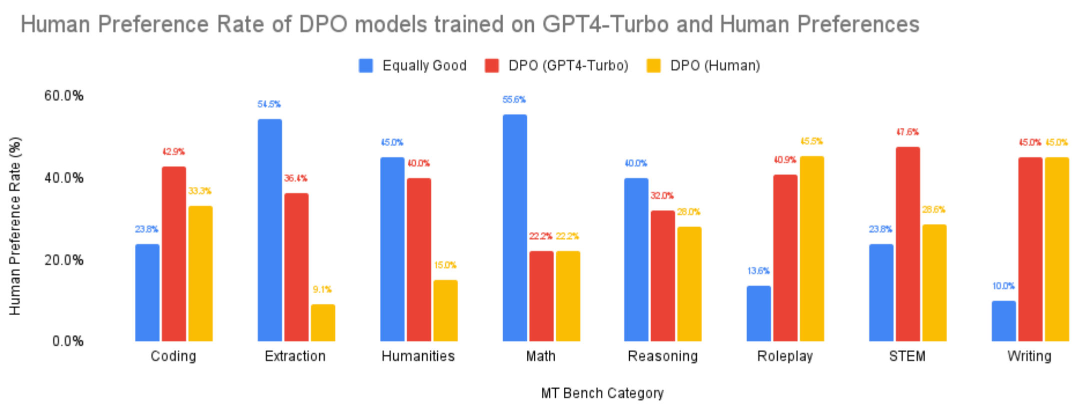


Figure 2: Analysis of Human Preference Rate by Domains in MT Bench: DPO Models trained on GPT4-Turbo vs. Human Preferences. Remarkably, the DPO model trained on GPT4-Turbo preferences demonstrates superior or comparable performance across all domains, with the exception of Roleplay.

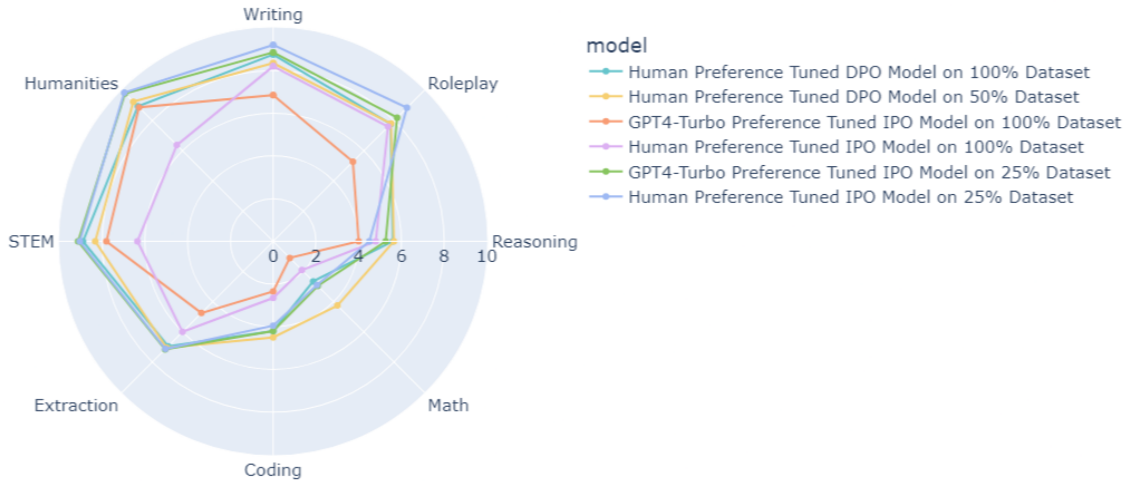


Figure 3: Analysis of GPT4 Ratings by Domains in MT Bench: DPO and IPO Models trained on different volumes of GPT4-Turbo and Human Preferences within the GAIR dataset. We present three comparisons that challenge the expected trends: IPO model trained on 25% versus 100% Human Preferences, IPO model trained on 25% versus 100% GPT4-Turbo Preferences, and DPO model trained on 50% versus 100% Human Preferences.

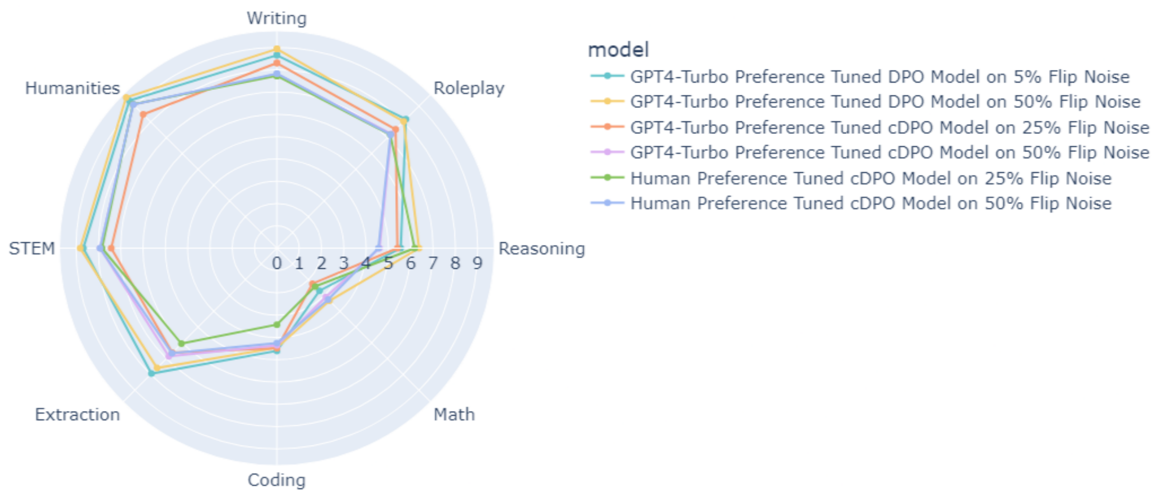


Figure 4: Analysis of GPT4 Ratings by Domains in MT Bench: DPO and cDPO Models trained on different proportions of flip noise induced in GPT4-Turbo and Human Preferences within the GAIR dataset. We present three noteworthy comparisons that challenge the expected trends: cDPO model trained on 25% versus 50% flip noise induced in Human Preferences, cDPO model trained on 25% versus 50% flip noise induced in GPT4-Turbo Preferences, and DPO model trained on 5% versus 50% flip noise induced in GPT4-Turbo Preferences.

Domain - GAIR (Train)	**Domain - MT Bench (Test)**	**# Samples - GAIR (Train)**
analyzing_general	Reasoning, Extraction, Writing, Roleplay	16
chitchat	Roleplay	239
code_correction_rewriting	Code	24
code_simplification	Code	1
counterfactual	Reasoning	52
explaining_code	Code	29
information_extraction	Extraction	30
keywords_extraction	Extraction	3
note_summarization	Extraction	1
question_generation	Reasoning	53
recommendation	Reasoning	45
solving_exam_question_with_math	Math	27
solving_exam_question_without_math	STEM, Humanities	39
text_simplification	Writing	7
text_to_text_translation	Writing	43
verifying_fact	Extraction	57
writing_cooking_recipe	Writing	47
writing_job_application	Writing	23
writing_marketing_materials	Writing	2
writing_personal_essay	Writing	29
writing_product_description	Writing	21
writing_social_media_post	Writing	10
writing_technical_document	Writing	13
creative_writing	Writing	275
instructional_rewriting	Writing	25
language_polishing	Writing	12
open_question	Writing	395
text_correction	Writing	14
title_generation	Writing	10
writing_advertisement	Writing	5
writing_email	Writing	79
writing_legal_document	Writing	17
writing_news_article	Writing	5
writing_presentation_script	Writing	12
writing_scientific_paper	Writing	6
writing_song_lyrics	Writing	41
functional_writing	Writing	195
paraphrasing	Writing	22
writing_blog_post	Writing	12
asking_how_to_question	Reasoning	100
classification_identification	Extraction	28
code_generation	Code	341
code_to_code_translation	Code	6
explaining_general	Reasoning	385
ranking	Reasoning	39
text_summarization	Extraction	93
brainstorming	Reasoning	165
data_analysis	Math	19
math_reasoning	Math, Reasoning	334
reading_comprehension	Reasoning	13
roleplay	Roleplay	131
value_judgement	Humanities	172
default	-	865
planning	Reasoning	75
seeking_advice	Roleplay	323

Table 6: Mapping between the domains represented in GAIR and MT Bench

Domain - MT Bench (Test)	# Samples - GAIR (Train)
Writing	1336
Reasoning	1277
Roleplay	924
Code	401
Math	380
Extraction	228
Humanities	211
STEM	39

Table 7: Data Volume in GAIR Corresponding to Domains in MT Bench

Data Volume (% Train Data)	Human	
	Loss = DPO (Tuned)	Loss = IPO (Tuned)
100%	7.184	7.113
75%	7.038	6.981
50%	7.181	6.722
25%	6.959	6.878

Table 8: Benchmarking performance of DPO and IPO models when trained with tuned β on different volumes of human preference data using MT Bench scores

Data Quality (% Flip Noise)	Human	
	Loss = DPO (Tuned)	Loss = cDPO (Tuned)
0%	7.184	7.184
5%	7.078	7.063

Table 9: Benchmarking performance of DPO and cDPO models when trained with tuned β on human preference datasets with different flip noise ratios using MT Bench scores

Algorithm	UltraFeedback	SHP
Baseline (SFT)	6.753	6.753
DPO	7.225	6.441

Table 10: Benchmarking DPO model performance on Ultrafeedback and SHP datasets using MT Bench scores

Data Volume (% Train Data)	UltraFeedback	SHP
100%	7.225	6.441
75%	7.419	6.438
50%	7.306	6.244
25%	7.384	6.122

Table 11: Benchmarking DPO model performance with varying sample proportions in Ultrafeedback and SHP datasets using MT Bench scores

Data Quality (% Flip Noise)	UltraFeedback	SHP
0%	7.225	6.441
5%	6.872	6.453
25%	6.691	6.272
50%	6.403	6.244
75%	5.928	5.664

Table 12: Benchmarking DPO model performance with varying flip noise in Ultrafeedback and SHP datasets using MT Bench scores

	UltraFeedback		SHP	
Data Volume (% Train Data)	Loss = DPO	Loss = IPO	Loss = DPO	Loss = IPO
100%	7.225	6.813	6.441	6.200
75%	7.419	5.853	6.438	6.469
50%	7.306	6.756	6.244	6.466
25%	7.384	6.344	6.122	6.419

Table 13: Benchmarking models with DPO and IPO loss functions across different Ultrafeedback and SHP dataset ratios using MT Bench scores

	UltraFeedback		SHP	
Data Quality (% Flip Noise)	**Loss = DPO**	**Loss = cDPO**	**Loss = DPO**	**Loss = cDPO**
0%	7.225	7.225	6.441	6.441
5%	6.872	6.484	6.453	6.425
20%	6.691	6.050	6.272	5.963
50%	6.403	6.325	6.244	6.403
75%	5.928	5.797	5.664	5.616

Table 14: Benchmarking models with DPO and cDPO loss functions across different Ultrafeedback and SHP flip noise ratios using MT Bench Scores.

Dataset	# Train	# Val	Domains / Sub-Datasets
Ultrafeedback	61135	2000	TruthfulQA, FalseQA, Evol-Instruct, UltraChat, ShareGPT
SHP	100000	1000	academia, anthropology, baking, carguys, culinary, docs, engineers, historians, hr, philosophy, physics, science, science fiction, social science, vet, change my view, explain like i'm five, legal advice
GAIR	5240	0	Chatbot Arena conversations
MT Bench	-	160	Writing, Roleplay, Extraction, Reasoning, Math, Coding, Knowledge I (STEM), and Knowledge II (humanities/social science)

Table 15: Overview of Datasets

RQ	Description	Text	Output Score
1	Prompt	You can see a beautiful red house to your left and a hypnotic greenhouse to your right, an attractive heated pink place in the front. So, where is the White House?	-
1	DPO Model trained on 100% of GPT4-Turbo preferences in GAIR dataset	The White House is not visible from this description as it is not mentioned. The description only describes other buildings and their colors.	3
1	DPO Model trained on 100% of Human preferences in GAIR dataset	The White House is likely located behind the red house and greenhouse, as it is the official residence and workplace of the President of the United States. The heated pink place in the front may be a separate building or structure on the same property.	2
1	Prompt	Benjamin went to a bookstore and purchased a variety of books. He bought 5 copies of a sci-fi novel, each priced at $20, 3 copies of a history book priced at $30 each, and 2 copies of a philosophy book for $45 each. What was the total cost of his purchases?	-
1	DPO Model trained on 100% of GPT4-Turbo preferences in GAIR dataset	The total cost of Benjamin's purchases is: 5 copies of sci-fi novel at $20 each = $100 3 copies of history book at $30 each = $90 2 copies of philosophy book at $45 each = $90 Total cost = $100 + $90 + $90 = $270	6
1	DPO Model trained on 100% of Human preferences in GAIR dataset	5 x $20 = $100 3 x $30 = $90 2 x $45 = $90 Total cost = $100 + $90 + $90 = $270 Therefore, the total cost of his purchases was $270.	4

Table 16: Examples of generated model outputs on MT Bench

The Ups and Downs of Large Language Model Inference with Vocabulary Trimming by Language Heuristics

Nikolay Bogoychev Pinzhen Chen Barry Haddow Alexandra Birch
School of Informatics, University of Edinburgh
{nbogoych,pinzhen.chen,bhaddow,a.birch}@ed.ac.uk

Abstract

Deploying large language models (LLMs) encounters challenges due to intensive computational and memory requirements. Our research examines vocabulary trimming (VT) inspired by restricting embedding entries to the language of interest to bolster time and memory efficiency. While such modifications have been proven effective in tasks like machine translation, tailoring them to LLMs demands specific modifications given the diverse nature of LLM applications. We apply two language heuristics to trim the full vocabulary—Unicode-based script filtering and corpus-based selection—to different LLM families and sizes. The methods are straightforward, interpretable, and easy to implement. It is found that VT reduces the memory usage of small models by nearly 50% and has an upper bound of 25% improvement in generation speed. Yet, we reveal the limitations of these methods in that they do not perform consistently well for each language with diminishing returns in larger models.

1 Introduction

Large language models (LLMs) are gaining increasing attention given their strong performance (Radford et al., 2019; Brown et al., 2020; Scao et al., 2022; Touvron et al., 2023). LLMs, especially multilingual ones, hold vocabulary items for many languages and scripts, which entail a costly matrix multiplication $H \times |V|$ in the output layer, where H is the hidden size and $|V|$ is the size of a vocabulary V. This expensive operation leads to increased costs of both memory and time given the autoregressive nature of LLM decoding. Given their substantial size, this latency in inference significantly escalates the expense of LLM deployment.

In practice, creating a sub-vocabulary V' with $|V'| \ll |V|$ and only loading its corresponding embedding entries for inference seems favourable since most logits from the output layer do not affect the hypothesis token(s) at each time step. Vocabulary trimming (VT) has been actively explored in machine translation (often called *shortlisting*, Schwenk et al., 2007; Le et al., 2012; Devlin et al., 2014)—it computes token-level alignments and makes potential target tokens a sub-vocabulary. While anticipating certain limitations such as domain mismatch (Bogoychev and Chen, 2021; Domhan et al., 2022), vocabulary shortlisting in LLMs poses a fundamental challenge: often LLM outputs are variable and open-ended, complicating the determination of the required lexicons. Recent attempts at multilingual pre-trained models select tokens in a task's language (Abdaoui et al., 2020; Ushio et al., 2023). Nonetheless, research in this direction is still limited, especially in speed considerations.

We follow the idea of fitting vocabulary to the language of the downstream task. Specifically, We examine two strategies: *Unicode-based filtering* where vocabulary items are removed if they do not belong to the task language, and *corpus-based selection* where we record vocabulary hits from a large representative corpus. After experimenting with LLMs from two families of different sizes, we identify a good upper bound of memory reduction with several limitations and outlooks: 1) Unicode-based script filtering maintains quality for Latin-based languages but harms languages requiring code-mixing. 2) Corpus-based selection leads to fewer alterations but is less effective in reducing the embedding size. 3) Embeddings are proportionally smaller in larger models (with smaller vocabularies). Yet we argue that VT can be applied orthogonally to other efficiency methods like efficient attention, quantization, etc.

2 Language-Based Vocabulary Trimming

We explore two ways to prepare sub-vocabulary for LLMs, focusing on only retaining tokens relevant to the language being generated. We test a batched setting on the fly: we determine a sub-

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 148–153
June 20, 2024

vocabulary for an entire batch because creating the sub-vocabulary separately for each input is too expensive in practice. Furthermore, we always include all tokens appearing in the inputs.

Script-based filtering This is done by filtering token strings that fall out of a language's Unicode range—keeping tokens in the writing script of that language. It should be especially effective for languages operating on unique scripts, such as Armenian, Chinese, Korean, etc since it allows for concise vocabulary restriction. This method might be less practical if a writing system is shared among multiple languages (e.g. Cyrillic or Latin alphabets), because it would be infeasible to limit the lexicons to those solely belong to a specific language, resulting in a relatively large sub-vocabulary. Moreover, this method would strictly rule out code-mixed texts, emojis, etc, which are used in real-world communications.

Corpus-based selection Another way is to tokenize a representative corpus in the desired language in advance and use the vocabulary entries that have been recorded to build a sub-vocabulary. This method is non-exhaustive because we could miss rare but valid tokens or suffer from domain mismatch between the vocabulary selection corpus and the downstream tasks at inference time.

3 Experimental Setup

Languages and test sets We experimented on four languages: Bulgarian, Chinese, English, and Spanish, to offer distinct conditions that cover different degrees of writing script overlap, code-mixing, etc, with details in Appendix A. We sample 50 prompt questions from OpenAssistant (Köpf et al., 2023) which are then human-translated into test languages. We decode them with beam size 1.

Metrics We consider efficiency-quality trade-offs. In terms of speed, we report end-to-end time to decode the entire test, including model loading and embedding slicing. As a quality indicator, we count the chances a model fails to produce *the exact same output* (miss) with a full vocabulary and with VT. In addition, we report the BLEU and chrF of the VT output w.r.t. to the *original output* with the full vocabulary (not the reference). Note that there is no gold reference due to the open domain nature; we hence prefix the two string metrics with an "o-".

Large language models We experiment with instruction-tuned LLMs based on BLOOM at various sizes (Scao et al., 2022) as well as LLaMA-7B (Touvron et al., 2023). We adopt Chen et al. (2024)'s models fine-tuned on machine translations of the Alpaca dataset (Taori et al., 2023) to test for open domain question answering, which maximizes the difficulty for VT as explained earlier.

BLOOM is multilingual and explicitly supports English, Spanish, and Chinese, but not Bulgarian. Consequently, it has a sizeable vocabulary of 250K and is therefore a prime and tempting candidate to reduce vocabulary for a specific language during inference. We experiment with the 560M, 1B7, and 7B1 checkpoints, with diminishing computational burden on the embedding and output layers.

LLaMA is an English-centric LLM with a small 32K vocabulary. We might have reduced benefit from VT because a proportionally lower amount of computation occurs in the output layer. On the other hand, since the LLM is European language-focused, we expect drastic vocabulary reductions compared to BLOOM for Bulgarian and Chinese.

Vocabulary trimming details We tokenize the test prompts and always include input tokens in the sub-vocabulary. We then apply either of the proposed selection methods. Script-based filtering checks whether a vocabulary entry belongs to a Unicode subset: Cyrillic for Bulgarian, ASCII for English, Latin Extended-A for Spanish, and Chinese characters for Chinese. Whereas for corpus-based selection, we tokenize a subset of WikiMatrix (Schwenk et al., 2021) containing Wikipedia texts for each language and record vocabulary hits.

For both selection methods, we keep the first 300 vocabulary entries of each LLM too, as those usually correspond to special tokens, Unicode bytes (for byte-level BPE), numbers, etc. We compute the sub-vocabulary offline and we do not record the time spent on pre-tokenizing a large corpus or extracting a Unicode subset in the measurements, as once done, these can be reused for every batch during inference. Script-based filtering takes under 60 seconds and corpus-based selection takes up to 10 minutes. Adding the inputs' tokens to the sub-vocabulary takes negligible time.

Hardware We conduct experiments both on CPU and GPU devices. For the CPU tests, we use Xeon Gold 6248 (40 Cores, 80 Threads), and for GPU tests, we use a single Nvidia RTX 3090. CPU in-

Language	$\lvert V\rvert$	BLOOM-560M				BLOOM-1B7				BLOOM-7B1			
		time	miss	o-BLEU	o-chrF	time	miss	o-BLEU	o-chrF	time	miss	o-BLEU	o-chrF
bg full	250680	05:26	–			15:18	–			65:01	–		
Unicode	22912	04:39	1	99.04	99.73	13:44	4	91.67	96.36	51:46	10	83.42	87.03
corpus	58642	04:49	0			09:34	1	99.49	99.85	60:28	3	91.68	95.84
oracle	1408	04:22	0			12:31	0			61:06	0		
en full	250680	07:37	–			16:35	–			55:08	–		
Unicode	186752	07:40	1	98.21	98.68	16:05	0			58:18	0		
corpus	113024	07:00	1	99.22	99.45	15:08	3	96.59	98.88	54:20	2	98.91	99.40
oracle	4736	06:14	0			13:06	0			48:46	0		
es full	250680	05:58	–			12:26	–			63.15	–		
Unicode	187008	05:48	0			12:01	0			59:15	0		
corpus	112128	05:37	0			11:34	4	95.91	97.71	57:41	4	94.46	96.25
oracle	4736	04:53	0			09:26	0			51:43	0		
zh full	250680	06:29	–			15:27	–			55:09	–		
Unicode	51584	05:54	16	53.32	70.38	13:09	21	47.66	63.66	50:50	22	47.76	63.60
corpus	104320	06:08	11	66.17	78.44	14:08	16	63.26	73.99	46:39	17	62.37	76.55
oracle	4096	05:16	0			12:07	0			50:50	0		

Table 1: CPU VT results for the BLOOM family.

Language	$\lvert V\rvert$	LLaMA-7B			
		time	miss	o-BLEU	o-chrF
bg full	32000	117:15	–		
Unicode	4736	125:55	19	74.13	81.51
corpus	26496	132:24	5	97.75	98.44
oracle	2048	123:06	0		
en full	32000	113:52	–		
Unicode	27520	125:57	6	79.43	88.91
corpus	30720	111:30	19	93.07	97.14
oracle	4480	119:32	0		
es full	32000	131:03	–		
Unicode	27648	128:00	8	89.60	91.62
corpus	30336	129:26	2	97.08	99.15
oracle	3456	123:25	0		
zh full	32000	130:42	–		
Unicode	2688	114:39	13	75.43	83.36
corpus	28160	119:58	2	95.47	97.80
oracle	1536	126:16	0		

Table 2: CPU VT results for LLaMA-7B.

ference is performed in float32 precision, whereas GPU inference is in int8 (Dettmers et al., 2022).

4 CPU Results and Discussions

Upper bound performance First of all, we conduct an *oracle* vocabulary selection experiment to find the theoretical upper bound for speed and memory improvements: we run inference using the full vocabulary and we add the used vocabulary items to a *oracle* sub-vocabulary.

BLOOM versus LLaMA We present CPU results for the BLOOM family in Table 1 and those for LLaMA-7B in Table 2. We observe around 20% time improvements with the smaller BLOOM at 560M and 1B7, but only 5–10% in the 7B models. As the model grows in size, the oracle upper bound sees decreasing gains, due to the proportion of the embedding matrices becoming smaller in a larger model. By comparing BLOOM-7B1 with LLaMA-7B, we also find that the larger the base vocabulary, the more effective VT is. The oracle vocabulary is more than an order of magnitude smaller than our VT approaches, but in practice, it would be difficult to reduce the vocabulary size by as much.

Speed numbers of LLaMA-7B on CPU are relatively inconsistent and had wide variance across test runs. We attribute this to the small vocabulary size and thus less computational footprint in the output layer affected by VT. Also, there could be various scheduling issues and non-deterministic cache accesses as GEMM operations are split across the 40 cores of the CPU.

4.1 Script-based vocabulary trimming

When applying script-based filtering, we observe different trends in English and Spanish compared to Bulgarian and Chinese. For BLOOM, the sub-vocabulary size for Bulgarian and Chinese can be reduced to 10–20%, whereas for English and Spanish, it remains at 60%. This is potentially because BLOOM allocated more vocabulary items for European languages which are the dominant ones when the tokenizer is trained. Generally, the inference time reduces to between full and oracle vocabulary. In terms of misses, the model can maintain almost the same outputs with and without VT for English and Spanish. However, there are 10–20% misses for Bulgarian and 30–40% for Chinese.

LLaMA-7B results are less favourable: script-based filtering does not significantly reduce the vocabulary size for English and Spanish, and all languages suffer from relatively high misses between 10–40%. Specifically for Bulgarian and Chinese, we argue that Unicode filtering could be too harsh as sometimes English characters are code-mixed in the language and cannot be avoided, e.g., when generating a website link. Therefore, we conclude that VT based on the writing script can improve inference efficiency without degrading performance for a multilingual LLM to generate Latin languages, but it is less feasible for non-Latin languages or English-centric LLMs with a smaller vocabulary.

4.2 Corpus-based vocabulary trimming

Corpus-based selection leaves a much larger vocabulary for Bulgarian and Chinese but reduces the vocabulary to half or less for English and Spanish. This method produces a more balanced sub-vocabulary for each language likely due to the inclusion of tokens outside of the desired language. However, for LLaMA-7B which has a small vocabulary in the first place, this approach keeps most of the entries for all languages and is thus not useful.

The corpus-based selection also ameliorates the quality problem to some extent by allowing for code-mixing (usually English), although the Chinese VT models still struggle to produce identical output as the full vocabulary models. Overall, we see a small but consistent reduction in runtime with BLOOM for this VT approach, indicating its practicality at least for English.

4.3 Memory

Besides speed considerations, VT can lead to ample memory footprint reduction, especially for smaller models like BLOOM-560M, where the model size is dominated by the vocabulary (nearly 50% of all model parameters). In practice, these models are small enough to fit in modern GPUs and CPUs, so the reduced memory is not game-changing. On the other hand, when looking at bigger models like BLOOM-7B1 or LLaMA-7B, vocabulary makes up just a tiny portion of the overall number of parameters and thus the relative reduction in model size is modest and could not enable the use of smaller GPUs. We can view this as a proxy judgement about the computational distribution of the model: The larger the model, the less time is spent in the output layer, and thus the smaller the impact of VT is. Exact memory numbers are available in Table 3.

Language	BLOOM				LLaMA	
	$\|V\|$	560M	1B7	7B1	$\|V\|$	7B
Full model	250680	2.10	6.10	27.10	32000	27.10
Embedding matrix or output layer						
full vocab	250680	0.90	1.90	3.80	32000	0.50
bg Unicode	22912	0.09	0.18	0.36	4736	0.07
bg corpus	58642	0.22	0.45	0.90	26496	0.41
en Unicode	186752	0.70	1.40	2.80	27520	0.43
en corpus	113024	0.44	0.88	1.70	30720	0.48
es Unicode	187008	0.70	1.40	2.80	27648	0.43
es corpus	112128	0.43	0.86	1.70	30336	0.47
zh Unicode	51584	0.20	0.40	0.80	2688	0.04
zh corpus	104320	0.40	0.80	1.60	28160	0.44

Table 3: Theoretical memory footprint (in GB) for BLOOM and LLaMA with float32 featuring the embedding matrix.

5 GPU results

In addition to CPU tests, we performed the same BLOOM experiments on a GPU and observed that all three selection criteria including the oracle do not lead to improved inference speed. Small performance differences might amount to little more than noise when the overhead of model slicing is considered. We hypothesize that GPUs are designed for multiplying large matrices, so reducing the matrix size, even to the extremity of an oracle sub-vocabulary, is not able to offer any speedup. This is consistent with Bogoychev et al. (2020)'s findings in applying shortlists to neural machine translation on GPUs. We list GPU results for BLOOM in Appendix B Table 4.

6 Conclusion

We presented a study of two straightforward language-inspired vocabulary trimming methods to speed up inference and save memory for large language model deployment. Experiments reveal ups and downs. While we can achieve speed improvements, it does not guarantee that the output is not altered compared to full vocabulary generation. With the models tested, we see the feasibility of our proposed approaches for English and Spanish, but there are shortcomings when considering languages written in non-Latin script and requiring code-mixing. In terms of efficiency, the reduction in inference time is less pronounced compared with memory saving.

Ethical Considerations

Our study aimed solely at reducing the computational resource consumption for deploying large language models. Our analysis contributes to the understanding of language heuristics in trimming an LLM vocabulary. While there is minimal risk associated with generating harmful content, it is no different for other research on large language models. We believe research into this direction has a positive impact in terms of energy saving and service deployment.

Acknowledgement

This work has received funding from UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant numbers 10052546 and 10039436].

References

Amine Abdaoui, Camille Pradel, and Grégoire Sigel. 2020. Load what you need: Smaller versions of mutililingual BERT. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*.

Nikolay Bogoychev and Pinzhen Chen. 2021. The highs and lows of simple lexical domain adaptation approaches for neural machine translation. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*.

Nikolay Bogoychev, Roman Grundkiewicz, Alham Fikri Aji, Maximiliana Behnke, Kenneth Heafield, Sidharth Kashyap, Emmanouil-Ioannis Farsarakis, and Mateusz Chudyk. 2020. Edinburgh's submissions to the 2020 machine translation efficiency task. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*.

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. In *Advances in Neural Information Processing Systems*.

Pinzhen Chen, Shaoxiong Ji, Nikolay Bogoychev, Andrey Kutuzov, Barry Haddow, and Kenneth Heafield. 2024. Monolingual or multilingual instruction tuning: Which makes a better alpaca. In *Findings of the Association for Computational Linguistics: EACL 2024*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm.int8(): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Tobias Domhan, Eva Hasler, Ke Tran, Sony Trenous, Bill Byrne, and Felix Hieber. 2022. The devil is in the details: On the pitfalls of vocabulary selection in neural machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagyfi, et al. 2023. Openassistant conversations - democratizing large language model alignment. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Hai Son Le, Alexandre Allauzen, and François Yvon. 2012. Continuous space translation models with neural networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

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

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176B-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

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 the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*.

Holger Schwenk, Marta R. Costa-jussà, and Jose A. R. Fonollosa. 2007. Smooth bilingual n-gram translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. GitHub repository.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Asahi Ushio, Yi Zhou, and Jose Camacho-Collados. 2023. Efficient multilingual language model compression through vocabulary trimming. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.

A Languages

We experimented with Bulgarian, Chinese, English, and Spanish, to cover different conditions and use cases regarding writing scripts and text usage. English and Spanish use the same script and have a high overlap in vocabulary with many other languages after tokenization. Since LLMs are English-centric, we examine how effective of a sub-vocabulary we can find when it is not possible to shortlist merely based on the script. Bulgarian is a low-resource language written in the Cyrillic script. Most multilingual language models have lower amounts of Cyrillic tokens, so we expect that script-based filtering will leave a small sub-vocabulary; however, since Cyrillic is used by other languages, we will inevitably end up with vocabulary items that do not belong to Bulgarian. Finally, Chinese is a high-resource language with a unique script; Unicode filtering would be the most effective in this case.

B GPU Performance

We provide GPU performance numbers in Table 4. Unfortunately, neither of the language-based vocabulary trimming methods can improve time efficiency.

Language	$\|V\|$	BLOOM -560M		BLOOM -1B7		BLOOM -7B1	
		time	miss	time	miss	time	miss
bg full	250680	05:22	–	08:29	–	14:43	–
Unicode	22912	05:23	0	08:45	6	14:35	17
corpus	58642	05:22	0	08:38	1	14:33	10
oracle	1408	05:21	0	09:06	0	14:33	0
en full	250680	06:50	–	09:02	–	11:54	–
Unicode	186752	06:54	0	08:52	0	11:46	0
corpus	113024	06:38	2	08:56	3	11:59	3
oracle	4736	06:43	0	09:00	0	11:52	0
es full	250680	06:17	–	07:05	–	12:35	–
Unicode	187008	06:13	0	07:03	0	12:17	0
corpus	112128	06:15	1	7:10	3	12:30	3
oracle	4736	06:26	0	07:23	0	12:17	0
zh full	250680	05:37	–	08:47	–	11:58	–
Unicode	51584	06:10	15	08:34	20	11:22	29
corpus	104320	06:03	11	09:01	16	11:35	13
oracle	4096	05:35	0	08:42	0	11:46	0

Table 4: GPU VT results for the BLOOM family.

Multi-Task Learning with Adapters for Plausibility Prediction: Bridging the Gap or Falling into the Trenches?

Annerose Eichel and Sabine Schulte im Walde
Institute for Natural Language Processing, University of Stuttgart
{annerose.eichel,schulte}@ims.uni-stuttgart.de

Abstract

We present a multi-task learning approach to predicting semantic plausibility by leveraging 50+ adapters categorized into 17 tasks within an efficient training framework. Across four plausibility datasets in English of varying size and linguistic constructions, we compare how models provided with knowledge from a range of NLP tasks perform in contrast to models without external information. Our results show that plausibility prediction benefits from complementary knowledge (e.g., provided by syntactic tasks) are significant but non-substantial, while performance may be hurt when injecting knowledge from an unsuitable task. Similarly important, we find that knowledge transfer may be hindered by class imbalance, and demonstrate the positive yet minor effect of balancing training data, even at the expense of size.

1 Introduction

The ability to distinguish between plausible and implausible events represents a crucial building block for natural language processing (NLP). While existing models include classical transformer-based approaches (Porada et al., 2019; Emami et al., 2021), researchers also devise world-knowledge features (Wang et al., 2018), and examine lexical abstraction chains (Porada et al., 2021) in order to integrate relevant but yet missing information. In contrast, our work tackles the prediction of plausibility from a novel perspective, by testing whether knowledge from different tasks may be used to fill knowledge gaps and to improve plausibility models in low- to mid-size resource scenarios. Leveraging adapters (Pfeiffer et al., 2020a, 2021; Poth et al., 2023) as an efficient multi-task learning framework, we train 53 task adapters categorized into 17 tasks ranging from syntactic problems such as parsing to lexical semantics tasks such as abstractness prediction as well as sentence- and discourse-level semantics problems such as question answering. Across four plausibility datasets in English of varying size and linguistic constructions, we compare how models perform without external information (single-task adapters) in contrast to models provided with knowledge from other tasks (multi-task learning with adapter-fusion). In particular, the main goal of this paper is not to improve state-of-the-art results for each dataset but to explore whether task transfer through adapter-fusions works better than single-task adapters. More specifically, we are interested in the relationships between the source tasks (e.g., abstractness prediction or parsing) and the target task (plausibility prediction), and investigate which kind of knowledge is potentially relevant but yet missing for successfully predicting whether a given event is plausible or implausible. We first train single-task adapters for plausibility using each datasets' training data, and then explore the impact of additional within-task data regarding class balance as a potential factor. This is relevant insofar as language models (LMs) are commonly pretrained on mainly plausible training data and should thus be expected to perform better for plausible than implausible data. In a second step, we train and evaluate a range of adapter fusion models.

Our results indicate that (i) depending on the dataset, single-task adapter models can represent a viable alternative to full fine-tuning, (ii) knowledge from different tasks does not substantially improve and even hurt performance, depending on task, dataset, and training data setting, and (iii) adding in-domain data and removing class imbalance sustains plausibility prediction across datasets. Analyzing task categories reveals minimum negative impact from syntactic tasks, followed by discourse-level and lexical semantics tasks. We thus conclude that given the prerequisite of class balance, knowledge transfer through adapter fusion does not lead to substantial improvements for plausibility prediction when leveraging complementary tasks and might be even hurt in case of more closely related tasks.

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 154–168
June 20, 2024 ©2024 Association for Computational Linguistics

2 Background and Related Work

Modeling Semantic Plausibility While classical distributional models tend to model selectional preferences rather than semantic plausibility (Erk et al., 2010), there has been a line of advances to model plausibility (Wang et al., 2018; Porada et al., 2019; Pyatkin et al., 2021; Tang et al., 2023), including approaches to inject or induce knowledge at various levels.[1] For example, Wang et al. (2018) enhance a neural classifier to make use of manually annotated world-knowledge features of subjects and objects in (im)plausible events, and substantially increase performance. Porada et al. (2021) explore a transformer-based approach and show that providing abstractions over subjects and objects in form of lexical hierarchies is not sufficient to boost performance over a vanilla RoBERTa model. Emami et al. (2021) explore the effect of adjectival modifiers on event plausibility with transformers, and demonstrate that neither the adjective itself nor taxonomic classes help in correctly determining plausibility. More recently, Bang et al. (2023) consider a larger model and report results on physical semantic plausibility using ChatGPT with a prompting approach on a PEP-3K (Wang et al., 2018) sample. However, the strength of the presented findings is limited by their focus on only 30 of the available 3,062 *s-v-o* events ($\leqslant 1\%$). Providing insights from a slightly different perspective, Liu et al. (2023) devise a model to estimate the plausibility of commonsense statements with the goal of verification. Leveraging commonsense QA datasets and knowledge bases to substantially scale up training data and experimenting with different training objectives, they show that more data and a larger model (T5-XXL) significantly improve performance on commonsense verification.

In our work, we address the challenge of modelling plausibility from a novel angle, and test whether leveraging knowledge from other tasks improves plausibility prediction with a standard-sized transformer model through providing information from closely related vs. vastly different tasks, thus exploring which knowledge gaps need to be filled.

Multi-Task Learning with Adapters Adapters (Houlsby et al., 2019) have been introduced as a parameter-efficient fine-tuning approach[2] for transformers (Vaswani et al., 2017) with comparable performance. They consist of sets of additional task-specific parameters that are introduced at every layer of a transformer and updated during fine-tuning, while the remaining PLM parameters are kept frozen. Since adapters can be used in a modular fashion, they are particularly well-suited for multi-task and cross-lingual transfer learning (He et al., 2021; Pfeiffer et al., 2020b, 2021; Ansell et al., 2021) as well as to inject external knowledge sources to solve downstream tasks (Lauscher et al., 2021; Falk and Lapesa, 2023).

We use *Adapters* (Pfeiffer et al., 2020a, 2021; Poth et al., 2023) as our framework; it enables both training task-specific adapters, i.e., knowledge extraction, and combining the trained adapters in a second step through knowledge composition in a non-destructive way.

3 Datasets

We harness four English datasets for plausibility:

PEP-3K (Wang et al., 2018) consists of 3,062 subject-verb-object events in English that focus on highly concrete concepts, e.g., *lion-destroy-house*. Events have been judged *plausible* or *implausible* by five crowd-sourced annotators.

20Q[3] comprises a collection of 20 question-style games played by crowd-sourced workers. One player tries to guess a topic by asking questions to the other player (who knows the topic) that lead to a discrete answer. Possible answers are {*always, usually, sometimes, rarely, never*}. We use the dataset version adapted for binary plausibility classification by Porada et al. (2021).

ADEPT (Emami et al., 2021) encompasses 16,115 English sentence pairs differing only in an adjective modifying a noun, e.g., {A horse goes away ↔ A *dead* horse goes away}. The dataset was collected for predicting changes in plausibility within a multi-class setting; the set of labels is {*impossible, less likely, equally likely, more likely, necessarily true*}. To train and evaluate on this dataset, we map every $s1$ from the sentence pairs $\langle s1, s2\rangle$ to the label *plausible*. For sentences $s2$ we map the labels *impossible* and *less likely* to *implausible*, and the labels *equally likely, more likely* and *necessarily true* to *plausible*.

ELLIE (Testa et al., 2023) is a small dataset composed of 575 English elliptical constructions, i.e., the dataset was constructed to evaluate the effect of

[1] For a brief discussion wrt. the distinction between selectional preference and semantic plausibility, we refer to App. A.

[2] For an overview of different adapter architectures, we refer to Pfeiffer et al. (2024).

[3] https://github.com/allenai/twentyquestions

argument thematic fit when resolving ellipses. Instances are labeled *typical*, *atypical*, or *violating selectional preference* regarding agents and patients. We map the labels *typical* and *atypical* to *plausible*, and instances *violating selectional preference* to *implausible*. While we add ELLie data for training, our main use of the dataset is for in-domain evaluation, to assess generalization to complex linguistic constructions.

For an overview of dataset statistics, training setting sizes, dataset splits, and details regarding the conversion of selectional preference datasets such as ELLIE for plausibility modeling, we refer to App. B.

4 Models

Single-Task Adapters To establish baseline performance for predicting plausibility without knowledge from additional tasks, we train **single-task (ST) adapters**.[4] To further explore the influence of adding within-task knowledge and class imbalance, we experiment with training (i) on the train portion of each *target* dataset (TRAIN); (ii) on all full datasets except for the *target* dataset, and evaluate on the *target* datasets' dev and test, with and without removing class imbalance (W/O TRAIN, W/O TRAIN+B); (iii) on all datasets, including train of the *target* dataset, and evaluate on *target* datasets' dev and test, with and without removing class imbalance (W/ TRAIN, W/ TRAIN+B). To compare results to previous work, we test models trained on the respective other datasets (W/O TRAIN, W/O TRAIN+B) and evaluate on PEP-3K and 20Q dev and test set splits as used by Porada et al. (2021).

We conduct an intermediate error analysis on our ST adapter models, in order to understand how training data choices influence model performance. For this, we calculate error overlap at instance level and compute Spearman's ρ across training settings. In case of substantial overlap between wrongly predicted instances, we assume low influence of training data. In the reverse case, we assume that training data does make a difference. More details and results are presented in App. C, Fig. 1 with observations indicating that additional training data leads to different types of errors and may thus add relevant knowledge. Furthermore, removing class imbalance alters sets of errors significantly, in case of a previously imbalanced dataset.

[4] `https://github.com/AnneroseEichel/adapters-for-pp`

Adapter Fusion We make use of 53 **source-task adapters** trained on 17 tasks categorized into syntactic, lexical-semantic, and sentence/discourse level (for an overview see Table 5 in App. C). Whenever available, we harness existing adapter implementations via `adapterhub` or `huggingface`. We train two task adapters, with different motivations: (i) we predict a selectionally preferred argument using the SP-10K dataset (Zhang et al., 2019), because we are interested in the impact of adapters trained on the closely related task of selectional preference prediction; and (ii) we predict a word's abstractness score using a modified version of the concreteness norms by Brysbaert et al. (2014), because event abstractness vs. concreteness is potentially correlated with semantic plausibility (Eichel and Schulte im Walde, 2023).

To incorporate knowledge from other tasks, we train task-based **adapter fusions** using all task adapters belonging to a task, plus a task adapter for plausibility prediction.

Experimental Setup We use RoBERTa (Liu et al., 2019) (`roberta-base`) as the backbone transformer for all models. We **train ST adapters** for our target task of predicting whether a text input is plausible or not by using a task-specific prediction head, thus following the training setup recommended by Poth et al. (2023). We pick the best model based on development set results optimizing for macro F1. To **train adapter fusions**, we use the three best-performing target task adapters based on ST performance. We consider three training data settings to explore knowledge transfer (i) in low-resource settings and high class imbalance (TRAIN), (ii) in cases where no train portion might be available or included (W/O TRAIN), and (iii) for balanced datasets (W/ TRAIN+B). Our hypothesis is that training with small and imbalanced datasets may particularly benefit from knowledge transfer. The training setup mirrors the single-task setup, except for using a smaller learning rate and a larger batch size as in Poth et al. (2023), with models optimized for macro F1. For more details, we refer to App. D.

5 Results

In the following, we present our results comparing fusion-based against single-task adapter models for the target task of assessing plausibility. We use the Almost Stochastic Order (ASO) test (Del Barrio et al., 2018; Dror et al., 2019) as implemented by Ulmer et al. (2022) to assess which training and

BL/tasks	PEP-3K train	PEP-3K w/o train	PEP-3K w/ train+b	20Q train	20Q w/o train	20Q w/ train+b	ADEPT train	ADEPT w/o train	ADEPT w/ train+b
ST	0.80	0.69	0.82	0.76	**0.66**	0.76	**0.76**	**0.57**	0.82
(Morpho-)Syntactic									
chunk	0.80	0.68	0.82	0.76	0.62	**0.77**	0.72	0.55	**0.83**
dep	0.79	0.67	0.81	0.77	0.62	**0.77**	**0.74**	0.55	**0.83**
ged	**0.81**	0.68	0.82	0.76	0.63	**0.78**	0.71	0.54	**0.83**
la	0.80	0.68	0.82	0.76	0.62	**0.77**	0.72	0.54	**0.83**
ner	**0.81**	0.68	0.82	0.76	0.62	**0.77**	0.71	0.55	**0.83**
parse	0.80	0.67	0.82	0.76	0.63	**0.77**	0.72	0.54	**0.83**
tag	0.79	0.67	0.81	0.76	0.63	**0.77**	0.71	0.56	**0.83**
Lexical Semantics									
abstr	0.79	0.68	0.81	0.77	0.62	**0.77**	**0.75**	0.55	**0.83**
emo	0.80	0.68	0.82	0.76	0.63	**0.77**	0.71	0.55	**0.83**
senti	0.80	0.68	0.82	0.76	0.62	**0.77**	0.73	0.55	**0.83**
sp	0.80	0.68	0.81	0.77	0.63	0.76	0.71	0.55	**0.83**
Sentence/Discourse-level Semantics									
arg	0.80	0.67	0.82	0.76	0.62	**0.77**	0.72	0.54	**0.83**
csr	0.78	0.67	0.82	0.76	0.63	**0.77**	0.71	0.56	**0.83**
mrc	0.79	0.68	0.83	0.76	0.62	**0.78**	0.72	0.54	**0.83**
nli	**0.81**	0.66	0.81	0.75	0.62	**0.77**	0.70	0.55	**0.83**
qa	0.78	0.68	0.82	0.75	**0.64**	**0.77**	0.71	0.55	**0.83**
sts	0.80	0.68	0.81	0.76	0.63	**0.77**	0.72	0.56	**0.83**

Table 1: Performance of fusion models across datasets and training data settings, with test set performance reported using AUC averaged over three runs (see Table 4 for an overview including standard deviation). Performance is compared to the best-performing ST adapter models (cf. Table 2 for all ST adapter results). Orange and teal coloring refer to a decrease and increase in results, respectively, while gray coloring denotes similar performance. Values in bold denote *Almost Stochastic Dominance* over other models in the same column ($\epsilon_{\min} < \tau$ with $\tau = 0.5$). While changes in performance are statistically significant, the absolute magnitude of performance increase and decrease remains within maximum +2% and -6%.

task setups are most successful at a statistically significant level. That is, we compare corresponding pairs of models based on three random seeds (5, 17, 42), each using ASO with a confidence level of $\alpha = 0.05$, before adjusting for all pair-wise comparisons using the Bonferroni correction.

Does knowledge transfer through adapter fusion improve models of plausibility? Table 1 presents our main results, comparing the multitude of fusion models against the best-performing single-task adapters. We observe a range of interesting insights: (i) Knowledge transfer does not lead to substantial performance gains in low-resource scenarios (PEP-3K, 20Q, train) across tasks from all categories. (ii) When training on other than the original training data, adding knowledge from different tasks either hurts in most cases (20Q, ADEPT, w/o train), or yields comparable results (PEP-3K), but does not explicitly help. (iii) When making use of as much balanced-out training data as possible, including representations from a different task either sustains (20Q, ADEPT plausibility prediction performance, train+b) or at least does not hurt the performance (PEP-3K). Regarding task categories, our study reveals minimum negative impact from syntactic tasks, closely followed by discourse-level tasks and (but with a larger margin) lexical-semantics tasks. We conclude that **given the prerequisite of class balance, plausibility prediction can be sustained but not substantially improved through complementary knowledge transfer in adapter fusion**, while more closely related tasks seem to rather hurt performance.

Does adding in-domain data improve models of plausibility? Table 2 looks into variants of our baseline single-task adapters with and without adding in-domain data. When training and evaluating on 20Q and ADEPT TRAIN, learning a combined representation including in-domain

Train Data	PEP-3K	20Q	ADEPT	PEP-3K-C	20Q-C	ELLIE
train	$0.80^{\pm 0.02}$	$\mathbf{0.76}^{\pm 0.01}$	$0.76^{\pm 0.01}$	-	-	-
w/o train	$0.69^{\pm 0.03}$	$0.66^{\pm 0.01}$	$0.57^{\pm 0.02}$	$\mathbf{0.68}^{\pm 0.00}$	$\mathbf{0.65}^{\pm 0.00}$	$0.50^{\pm 0.00}$
w/o train+b	$0.62^{\pm 0.03}$	$0.64^{\pm 0.02}$	$0.55^{\pm 0.01}$	$0.64^{\pm 0.01}$	$0.62^{\pm 0.01}$	$0.50^{\pm 0.01}$
w/ train	$\mathbf{0.83}^{\pm 0.01}$	$\mathbf{0.76}^{\pm 0.01}$	$0.74^{\pm 0.02}$	-	-	-
w/ train+b	$\mathbf{0.82}^{\pm 0.01}$	$\mathbf{0.76}^{\pm 0.01}$	$\mathbf{0.82}^{\pm 0.01}$	-	-	-

Table 2: Target task adapter performance comparison across datasets and train data settings. PEP-3K-C and 20Q-C refer to dev and test splits as devised by Porada et al. (2019), cf App. B for further details. We report test set performance using AUC, averaged over 3 runs, with standard deviation. Using *Almost Stochastic Order* (ASO) testing, we determine almost stochastic dominant models ($\epsilon_{\min} < \tau$ with $\tau = 0.2$), marked in bold.

datasets yields competitive results and seems to help with both small (PEP-3K) and larger datasets (ADEPT). In comparison to previous work (Porada et al., 2021) performing full fine-tuning on an automatically extracted 3M train set, our single-task adapters are acceptable for 20Q (Porada et al. (2021): 0.74, ours: 0.65). For PEP-3K, the single-task adapters are outperformed by full fine-tuning on only in-domain data using `BERT-large` (Porada et al., 2019), while reaching performance comparable to full fine-tuning on `RoBERTa-base` with an automatically extracted 3M train set and enforced lexical abstraction consistency (Porada et al., 2021) (Porada et al. (2019): 0.89 accuracy, Porada et al. (2021): 0.67 AUC, ours: 0.68 AUC). Thus, based on our study settings, we conclude that **low-resource plausibility prediction is likely to benefit from more data disregarding any class imbalance**, which, however, decreases with growing dataset size.

6 Limitations and Future Directions

Events based on *s-v-o* events or comparably simple constructions have been successfully leveraged for exploring selection preference and thematic fit tasks (Erk et al., 2010; Zhang et al., 2019; Pedinotti et al., 2021). However, the addition of context could potentially resolve potential ambiguities in the *s-v-o* triples and thus improve plausibility prediction. Furthermore, while we train and evaluate our models on datasets such as ADEPT coming with sentence-level contexts, high class imbalance leads to a relatively small proportion of *implausible* sentences which are particularly relevant as LMs are usually pretrained on mostly plausible data and expected to inherently perform better for plausible expressions. We hope future research extends this work by collecting plausibility ratings for more complex constructions within broader contexts. Here, Liu et al. (2023) and Tang et al. (2023) present interesting work exploring the generation of implausible and less plausible but relevant outputs to complement their dataset with the goal of increasing model performance and assist humans in well-balanced decision-making, respectively.

Further, experiments with a wider variety of (larger) models represent a relevant future task to explore whether the presented negative results are specific to the used underlying transformer backbone or prevalent across model sizes and families.

Finally, in this work, we follow previous research (Wang et al., 2018; Porada et al., 2019, 2021) regarding the formulation of plausibility prediction as a binary classification task to discern *plausible* from *implausible* events. Plausibility can, however, also be captured in a graded way using more fine-grained labels that allow for graded classification such as the label set {*impossible, less likely, equally likely, more likely, necessarily true*} adopted by Emami et al. (2021) for modeling *change* in semantic plausibility between two sentences. We thus encourage further research on modeling plausibility from a graded perspective to capture the phenomenon at a more fine-grained level.

7 Conclusion

We tackled the task of discerning plausible from implausible events by adopting a multi-task learning perspective and exploring whether knowledge transfer from different tasks improves performance and reveals insights about relevant knowledge. Using 53 adapters categorized into 17 tasks, we found that complementary knowledge sustains but not substantially improves performance, while choosing a "wrong" task might seriously hurt the results. We further demonstrated that knowledge transfer may be hindered by class imbalance, and that balancing training data shows a significant positive yet non-substantial effect, even at the expense of size.

Ethics Statement

While humans excel at assessing plausibility, they might naturally disagree regarding the plausibility of an event such as *law-prohibit-discrimination*. In the course of the last decade, a growing line of research argues for the preservation and integration of disagreement in dataset construction, modelling, and evaluation (Aroyo and Welty, 2015; Pavlick and Kwiatkowski, 2019; Basile et al., 2021; Fornaciari et al., 2021; Uma et al., 2021). Automatically modeling plausibility thus bears the danger that what is considered plausible by a model will be closely related to what is represented as highly plausible in the existing datasets which do not capture disagreement in plausibility ratings. This might disadvantage certain assessments regarding the plausibility of an event or sentence that are so far underrepresented in the data. We therefore argue for the necessity to investigate how the presented or newly applied models process and handle data with potentially underrepresented perspectives on the plausibility of a given expression and to create more diverse plausibility datasets

Acknowledgements

We are grateful to the IMS SemRel research group for helpful suggestions and feedback regarding versions of this work. We would also like to thank the anonymous reviewers for their constructive feedback. Annerose Eichel received funding by the Hanns Seidel Foundation's Talent Program.

References

Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2019. ComQA: A community-sourced dataset for complex factoid question answering with paraphrase clusters. 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 307–317, Minneapolis, Minnesota. Association for Computational Linguistics.

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. 2017. The Parallel Meaning Bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 242–247, Valencia, Spain. Association for Computational Linguistics.

Alan Ansell, Edoardo Maria Ponti, Jonas Pfeiffer, Sebastian Ruder, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2021. MAD-G: Multilingual adapter generation for efficient cross-lingual transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4762–4781, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lora Aroyo and Chris Welty. 2015. Truth is a lie: Crowd truth and the seven myths of human annotation. *AI Magazine*, 36(1):15–24.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. 2023. A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–718, Nusa Dua, Bali. Association for Computational Linguistics.

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

Valerio Basile, Michael Fell, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, Massimo Poesio, and Alexandra Uma. 2021. We need to consider disagreement in evaluation. In *Proceedings of the 1st Workshop on Benchmarking: Past, Present and Future*, pages 15–21, Online. Association for Computational Linguistics.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. 2020. Abductive Commonsense Reasoning. In *International Conference on Learning Representations*.

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.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods*, 46(3):904–911.

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.

Ankush Chatterjee, Kedhar Nath Narahari, Meghana Joshi, and Puneet Agrawal. 2019. SemEval-2019 task 3: EmoContext contextual emotion detection in text. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 39–48, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Emmanuele Chersoni, Philippe Blache, and Alessandro Lenci. 2016. Towards a distributional model of semantic complexity. In *Proceedings of the Workshop on Computational Linguistics for Linguistic Complexity (CL4LC)*, pages 12–22, Osaka, Japan. The COLING 2016 Organizing Committee.

Emmanuele Chersoni, Ludovica Pannitto, Enrico Santus, Alessandro Lenci, and Chu-Ren Huang. 2020. Are word embeddings really a bad fit for the estimation of thematic fit? In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5708–5713, Marseille, France. European Language Resources Association.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

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, Berlin, Heidelberg. Springer Berlin Heidelberg.

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. 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 5925–5932, Hong Kong, China. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The CommitmentBank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung*, 23(2):107–124.

Eustasio Del Barrio, Juan A Cuesta-Albertos, and Carlos Matrán. 2018. An optimal transportation approach for assessing almost stochastic order. In *The Mathematics of the Uncertain*, pages 33–44. Springer.

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.

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

Rotem Dror, Segev Shlomov, and Roi Reichart. 2019. Deep Dominance - How to Properly Compare Deep Neural Models. 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 2773–2785. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. 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 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Annerose Eichel and Sabine Schulte im Walde. 2023. A Dataset for Physical and Abstract Plausibility and Sources of Human Disagreement. In *Proceedings of the 17th Linguistic Annotation Workshop (LAW-XVII)*, pages 31–45, Toronto, Canada. Association for Computational Linguistics.

Ali Emami, Ian Porada, Alexandra Olteanu, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2021. ADEPT: An adjective-dependent plausibility task. 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 7117–7128, Online. Association for Computational Linguistics.

Katrin Erk, Sebastian Padó, and Ulrike Padó. 2010. A flexible, corpus-driven model of regular and inverse selectional preferences. *Computational Linguistics*, 36(4):723–763.

Neele Falk and Gabriella Lapesa. 2023. Bridging Argument Quality and Deliberative Quality Annotations with Adapters. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 2469–2488, Dubrovnik, Croatia. Association for Computational Linguistics.

Tommaso Fornaciari, Alexandra Uma, Silviu Paun, Barbara Plank, Dirk Hovy, and Massimo Poesio. 2021. Beyond black & white: Leveraging annotator disagreement via soft-label multi-task learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2591–2597, Online. Association for Computational Linguistics.

Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. SemEval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. 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 394–398, Montréal, Canada. Association for Computational Linguistics.

Clayton Greenberg, Vera Demberg, and Asad Sayeed. 2015a. Verb polysemy and frequency effects in thematic fit modeling. In *Proceedings of the 6th Workshop on Cognitive Modeling and Computational Linguistics*, pages 48–57, Denver, Colorado. Association for Computational Linguistics.

Clayton Greenberg, Asad Sayeed, and Vera Demberg. 2015b. Improving unsupervised vector-space thematic fit evaluation via role-filler prototype clustering. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 21–31, Denver, Colorado. Association for Computational Linguistics.

Ruidan He, Linlin Liu, Hai Ye, Qingyu Tan, Bosheng Ding, Liying Cheng, Jiawei Low, Lidong Bing, and Luo Si. 2021. On the effectiveness of adapter-based tuning for pretrained language model adaptation. 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 2208–2222, Online. 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*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. 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 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Carina Kauf, Emmanuele Chersoni, Alessandro Lenci, Evelina Fedorenko, and Anna A. Ivanova. 2024. Comparing plausibility estimates in base and instruction-tuned large language models.

Carina Kauf, Anna A Ivanova, Giulia Rambelli, Emmanuele Chersoni, Jingyuan Selena She, Zawad Chowdhury, Evelina Fedorenko, and Alessandro Lenci. 2023. Event knowledge in large language models: the gap between the impossible and the unlikely. *Cognitive Science*, 47(11):e13386.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of North American Chapter of the Association for Computational Linguistics (NAACL)*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. SciTaiL: A Textual Entailment Dataset from Science Question Answering. In *AAAI Conference on Artificial Intelligence*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Anne Lauscher, Tobias Lueken, and Goran Glavaš. 2021. Sustainable modular debiasing of language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4782–4797, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiacheng Liu, Wenya Wang, Dianzhuo Wang, Noah Smith, Yejin Choi, and Hannaneh Hajishirzi. 2023. Vera: A general-purpose plausibility estimation model for commonsense statements. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1264–1287, Singapore. 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. *CoRR*, abs/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.

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. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Yuval Marton and Asad Sayeed. 2022. Thematic fit bits: Annotation quality and quantity interplay for event participant representation. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5188–5197, Marseille, France. European Language Resources Association.

Eleni Metheniti, Tim Van de Cruys, and Nabil Hathout. 2020. How relevant are selectional preferences for transformer-based language models? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1266–1278, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Joakim Nivre, Daniel Zeman, Filip Ginter, and Francis Tyers. 2017. Universal Dependencies. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Tutorial Abstracts*, Valencia, Spain. Association for Computational Linguistics.

Ulrike Padó, Matthew Crocker, and Frank Keller. 2006. Modelling semantic role pausibility in human sentence processing. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 345–352, Trento, Italy. Association for Computational Linguistics.

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.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Paolo Pedinotti, Giulia Rambelli, Emmanuele Chersoni, Enrico Santus, Alessandro Lenci, and Philippe Blache. 2021. Did the cat drink the coffee? challenging transformers with generalized event knowledge. In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 1–11, Online. Association for Computational Linguistics.

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.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020a. 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.

Jonas Pfeiffer, Sebastian Ruder, Ivan Vulić, and Edoardo Maria Ponti. 2024. Modular Deep Learning.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020b. 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.

Ian Porada, Kaheer Suleman, and Jackie Chi Kit Cheung. 2019. Can a gorilla ride a camel? learning semantic plausibility from text. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 123–129, Hong Kong, China. Association for Computational Linguistics.

Ian Porada, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2021. Modeling event plausibility with consistent conceptual abstraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1732–1743, Online. Association for Computational Linguistics.

Clifton Poth, Hannah Sterz, Indraneil Paul, Sukannya Purkayastha, Leon Engländer, Timo Imhof, Ivan Vuli, Sebastian Ruder, Iryna Gurevych, and Jonas Pfeiffer. 2023. Adapters: A Unified Library for Parameter-Efficient and Modular Transfer Learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 149–160, Singapore. Association for Computational Linguistics.

Valentina Pyatkin, Shoval Sadde, Aynat Rubinstein, Paul Portner, and Reut Tsarfaty. 2021. The possible, the plausible, and the desirable: Event-based modality detection for language processing. 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 953–965, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *arXiv e-prints*, page arXiv:1606.05250.

Philip Stuart Resnik. 1993. *Selection and information: A class-based approach to lexical relationships*. Ph.D. thesis, University of Pennsylvania.

Anna Rogers, Olga Kovaleva, Matthew Downey, and Anna Rumshisky. 2020. Getting closer to AI complete question answering: A set of prerequisite real tasks. 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 8722–8731. AAAI Press.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693, Melbourne, Australia. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106.

Enrico Santus, Emmanuele Chersoni, Alessandro Lenci, and Philippe Blache. 2017. Measuring thematic fit with distributional feature overlap. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 648–658, Copenhagen, Denmark. Association for Computational Linguistics.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. 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 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Asad Sayeed, Clayton Greenberg, and Vera Demberg. 2016. Thematic fit evaluation: an aspect of selectional preferences. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 99–105, Berlin, Germany. 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.

Christian Stab, Tristan Miller, Pranav Rai, Benjamin Schiller, and Iryna Gurevych. 2018. ukp sentential argument mining corpus.

Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. 2019. QuaRTz: An open-domain dataset of qualitative relationship 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 5941–5946, Hong Kong, China. Association for Computational Linguistics.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. 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 641–651, 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.

Liyan Tang, Yifan Peng, Yanshan Wang, Ying Ding, Greg Durrett, and Justin Rousseau. 2023. Less Likely Brainstorming: Using Language Models to Generate Alternative Hypotheses. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12532–12555, Toronto, Canada. Association for Computational Linguistics.

Davide Testa, Emmanuele Chersoni, and Alessandro Lenci. 2023. We Understand Elliptical Sentences, and Language Models should Too: A New Dataset for Studying Ellipsis and its Interaction with Thematic Fit. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3340–3353, Toronto, Canada. Association for Computational Linguistics.

Ottokar Tilk, Vera Demberg, Asad Sayeed, Dietrich Klakow, and Stefan Thater. 2016. Event participant modelling with neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 171–182, Austin, Texas. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Dennis Ulmer, Christian Hardmeier, and Jes Frellsen. 2022. deep-significance: Easy and meaningful signifcance testing in the age of neural networks. ML Evaluation Standards Workshop at the Tenth International Conference on Learning Representations, ICLR 2022 ; Conference date: 25-04-2022 Through 29-04-2022.

Alexandra Uma, Tommaso Fornaciari, Anca Dumitrache, Tristan Miller, Jon Chamberlain, Barbara Plank, Edwin Simpson, and Massimo Poesio. 2021. SemEval-2021 task 12: Learning with disagreements. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 338–347, Online. 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.

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 *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Su Wang, Greg Durrett, and Katrin Erk. 2018. Modeling semantic plausibility by injecting world knowledge. 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 303–308, New Orleans, Louisiana. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents.

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

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, Portland, Oregon, USA. 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*.

Hongming Zhang, Hantian Ding, and Yangqiu Song. 2019. SP-10K: A large-scale evaluation set for selectional preference acquisition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 722–731, Florence, Italy. Association for Computational Linguistics.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension.

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.

A Selectional Preference and Semantic Plausibility

In this work, we follow a clear distinction between the notions of selectional preference and (semantic) plausibility established by previous work (Wang et al., 2018; Porada et al., 2019, 2021; Eichel and Schulte im Walde, 2023).

Selectional preference (or *thematic fit*) is concerned with the semantic preference of a predicate for taking an argument (Resnik, 1993; Erk et al., 2010), e.g., the relative preference of the verb *pour* for the noun *water* as its nominal object. Label sets commonly consist of the labels {*typical, atypical*} which are often interpreted as *plausible* and *implausible* as well as an additional label *selectional preference violation* for constructions violating the notion of selectional preference. Proposed approaches to modeling selection preference at the level of events and sentences include corpus-based methods (Padó et al., 2006; Erk et al., 2010), unsupervised vector-based approaches (Baroni and Lenci, 2010; Greenberg et al., 2015a,b; Sayeed et al., 2016; Chersoni et al., 2016; Santus et al., 2017; Chersoni et al., 2020), supervised neural networks (Tilk et al., 2016; Zhang et al., 2019; Marton and Sayeed, 2022), as well as transformer-based approaches (Metheniti et al., 2020; Pedinotti et al., 2021; Testa et al., 2023; Kauf et al., 2023, 2024).

In contrast to selectional preference, evaluations of semantic plausibility emphasizes the importance of treating what is *atypical but still plausible* as an instance of what might be actually plausible though not highly frequent or potentially novel. Hence, modeling approaches not only focus on correctly modeling what is typical as plausible but seek to also capture what is atypical yet still plausible as plausible (Wang et al., 2018; Porada et al., 2021). This also seems to be in line with human perception of plausibility which tends to place atypical yet plausible events on the side of plausibility as opposed to categorizing what is less frequent as atypical, and thus implausible (Eichel and Schulte im Walde, 2023).

B Dataset Test Sets and Splits

PEP-3K Wang et al. (2018) only provide a split into plausible and implausible events, while we split the data into balanced train, dev, and test sets. To compare to additional previous work, we employ a 50% dev and 50% test split by Porada et al. (2019) whenever possible (PEP-3K-C).

20Q[5] In our work, we use a dataset version adapted for binary plausibility classification by Porada et al. (2021). In addition to the provided 50% dev and 50% test splits, we split the data into train, dev, and test sets (20Q-C).

ADEPT The adapted ADEPT dataset consists of 32,230 individual sentences which we keep in the original (now double-sized) train, dev, and test set splits.

ELLIE While ELLIE was introduced to capture "[...] the effect of argument thematic fit in solving ellipsis and reconstructing the missing element" (Testa et al., 2023), our re-mapping of the labels *typical* and *atypical* to *plausible*, and instances *violating selectional preference* to *implausible* does not eliminate but rather highlight the distinction between selectional preference and semantic plausibility outlined in App. A. More specifically, the conversion introduces a different label set and a change in label distribution to allow the usage of the data to capture semantic plausibility.

Table 3 shows an overview of dataset sizes as well as training and test data statistics.

Concerning **licenses** of the used datasets, we note that Wang et al. (2018) do not provide a specific license for PEP-3K.[6] 20Q is licensed under the Apache-2.0 license.[7] The ADEPT dataset (Emami et al., 2021) is distributed under the CC BY-SA 3.0 license and includes data from work licensed under the Creative Commons Attribution-ShareAlike license CC BY-SA 4.0. ADEPT is accompanied by a dataset sheet. Testa et al. (2023) do not provide a specific license for the ELLIE dataset.[8] As far as we know, our use of the listed datasets is consistent with their intended use. Based on the accompanying publications, dataset descriptions, and data sheets no data was identified that violates anonymisation.

C Intermediate Error Analysis Results

We perform an error analysis to further understand how training data choices influence model performance. In case of substantial overlap between wrongly predicted instances, we assume low influence of training data. If the reverse is observed, training data makes a difference. For this, we retrieve all incorrectly predicted instances from the test predictions using the best-performing seed for each dataset. We calculate error overlap at instance

[5] https://github.com/allenai/twentyquestions

[6] https://github.com/suwangcompling/Modeling-Semantic-Plausibility-NAACL18

[7] cf. https://github.com/allenai/twentyquestions

[8] https://github.com/Caput97/ELLie-ellipsis_and_thematic_fit_with_LMs/tree/main

Setting	PEP-3K	20Q	ADEPT	ELLIE
		Training Data		
TRAIN	2,459	4,076	25,784	-
W/O TRAIN	37,901	35,867	8,733	35,867
W/O TRAIN+B	13,504	11,476	8,394	11,476
W/ TRAIN	40,350	39,943	34,517	-
W/ TRAIN+B	15,953	15,552	14,926	-
		Dev Data		
DEV SET	306	510	3,222	-
(Porada et al., 2019)	1,531	2,548	-	-
		Test Data		
TEST SET	307	510	3,224	575
(Porada et al., 2019)	1,531	2,548	-	-

Table 3: Overview of dataset sizes where TRAIN denotes training on the train split of a specific dataset only, W/O TRAIN refers to training on the full size of all but a specific dataset, and W/ TRAIN settings include the full size of all but a specific dataset plus the train portion of a specific dataset. +B refers to a setting where class labels are balanced out, using the maximum number of implausible labels and a randomly drawn sample from possible plausible labels.

level and compute Spearman's ρ across training settings. Results are presented in Fig. 1 with our observations as follows: Firstly, training on all but a given dataset's train set vs. including a dataset's train set leads to a clearly distinct set of incorrectly predicted instances, with stronger correlations observed for ADEPT than for PEP-3K and 20Q. Secondly, removing class imbalance alters error sets more strongly for ADEPT ($\rho = 0.3$) than for PEP-3K and 20Q ($\rho = 0.6$) where datasets are already balanced out. This might also be the reason for the outlier observed for the high overlap between ADEPT's W/O TRAIN and W/O TRAIN+B.

D Experimental Details

As a RoBERTa model, we use the `roberta-base` implementation from `huggingface` (Wolf et al., 2020) that comes with 125M parameters. We leverage `Adapters` (Poth et al., 2023) as multi-task learning framework. Existing task adapters are harnessed through `adapterhub.ml/` and listed in Table 5, with paths to the source. We use `scikit-learn` (Pedregosa et al., 2011) to calculate metrics. For all experiments, including obtaining predictions from the various models, we use a single NVIDIA RTX A600 GPU.

E Single-Task Adapter Results Details

We show results comparing single-task adapters for the target task of assessing plausibility in Table 2. Table 4 presents the results comparing single task source and target adapters with fusion-based models. For both single-task and adapter-fusion results we report mean and standard deviation of AUC score, averaged over three runs. Single-task adapters reach good results when tested on a given dataset's own test set. When evaluated on data that has not been seen in the test set, we observe comparable and acceptable performance for similar linguistic constructions (PEP-3K-C and 20Q-C) where models are trained on in-domain data (e.g., PEP-3K, ADEPT, ELLIE) and evaluated on 20Q-C dev and test sets. However, when evaluating on ELLIE which consists of more complex linguistic constructions, performance drops to random chance, indicating that the model cannot make use of information learned during training.

Figure 1: Analysis of error overlap across training settings at instance level where Spearman's $\rho = 1$ and $\rho = -1$ indicate perfect and no overlap, respectively.

	PEP-3K			20Q			ADEPT		
tasks	**train**	**w/o train**	**w/ train+b**	**train**	**w/o train**	**w/ train+b**	**train**	**w/o train**	**w/ train+b**
ST	0.80 ± 0.02	0.69 ± 0.03	0.82 ± 0.01	0.76 ± 0.01	0.66 ± 0.01	0.76 ± 0.01	0.76 ± 0.01	0.57 ± 0.02	0.82 ± 0.01
(Morpho-)Syntactic									
chunk	0.80 ± 0.01	0.68 ± 0.00	0.82 ± 0.01	0.76 ± 0.01	0.62 ± 0.01	0.77 ± 0.02	0.72 ± 0.02	0.55 ± 0.02	0.83 ± 0.00
dep	0.79 ± 0.02	0.67 ± 0.00	0.81 ± 0.02	0.77 ± 0.02	0.62 ± 0.01	0.77 ± 0.01	0.74 ± 0.03	0.55 ± 0.00	0.83 ± 0.00
ged	0.81 ± 0.01	0.68 ± 0.02	0.82 ± 0.01	0.76 ± 0.01	0.63 ± 0.01	0.78 ± 0.01	0.71 ± 0.03	0.54 ± 0.01	0.83 ± 0.00
la	0.80 ± 0.01	0.68 ± 0.01	0.82 ± 0.01	0.76 ± 0.02	0.62 ± 0.01	0.77 ± 0.00	0.72 ± 0.04	0.54 ± 0.01	0.83 ± 0.00
ner	0.81 ± 0.01	0.68 ± 0.01	0.82 ± 0.00	0.76 ± 0.01	0.62 ± 0.00	0.77 ± 0.01	0.71 ± 0.03	0.55 ± 0.02	0.83 ± 0.01
parse	0.80 ± 0.01	0.67 ± 0.01	0.82 ± 0.00	0.76 ± 0.01	0.63 ± 0.01	0.77 ± 0.01	0.72 ± 0.02	0.54 ± 0.01	0.83 ± 0.00
tag	0.79 ± 0.01	0.67 ± 0.02	0.81 ± 0.00	0.76 ± 0.02	0.63 ± 0.00	0.77 ± 0.00	0.71 ± 0.00	0.56 ± 0.01	0.83 ± 0.00
Lexical Semantics									
abstr	0.79 ± 0.02	0.68 ± 0.01	0.81 ± 0.01	0.77 ± 0.02	0.62 ± 0.01	0.77 ± 0.01	0.75 ± 0.05	0.55 ± 0.01	0.83 ± 0.00
emo	0.80 ± 0.01	0.68 ± 0.01	0.82 ± 0.00	0.76 ± 0.01	0.63 ± 0.01	0.77 ± 0.02	0.71 ± 0.03	0.55 ± 0.01	0.83 ± 0.00
senti	0.80 ± 0.02	0.68 ± 0.01	0.82 ± 0.01	0.76 ± 0.02	0.62 ± 0.01	0.77 ± 0.00	0.73 ± 0.03	0.55 ± 0.02	0.83 ± 0.01
sp	0.80 ± 0.02	0.68 ± 0.00	0.81 ± 0.01	0.77 ± 0.01	0.63 ± 0.01	0.76 ± 0.01	0.71 ± 0.01	0.55 ± 0.01	0.83 ± 0.00
Sentence+Discourse-level Semantics									
arg	0.80 ± 0.01	0.67 ± 0.01	0.82 ± 0.01	0.76 ± 0.01	0.62 ± 0.01	0.77 ± 0.01	0.72 ± 0.03	0.54 ± 0.01	0.83 ± 0.00
csr	0.78 ± 0.03	0.67 ± 0.00	0.82 ± 0.01	0.76 ± 0.02	0.63 ± 0.00	0.77 ± 0.01	0.71 ± 0.03	0.56 ± 0.01	0.83 ± 0.01
mrc	0.79 ± 0.01	0.68 ± 0.02	0.83 ± 0.00	0.76 ± 0.01	0.62 ± 0.01	0.78 ± 0.01	0.72 ± 0.02	0.54 ± 0.01	0.83 ± 0.01
nli	0.81 ± 0.01	0.66 ± 0.01	0.81 ± 0.01	0.75 ± 0.01	0.62 ± 0.02	0.77 ± 0.01	0.70 ± 0.02	0.55 ± 0.01	0.83 ± 0.01
qa	0.78 ± 0.02	0.68 ± 0.02	0.82 ± 0.01	0.75 ± 0.03	0.64 ± 0.01	0.77 ± 0.00	0.71 ± 0.03	0.55 ± 0.01	0.83 ± 0.01
sts	0.80 ± 0.02	0.68 ± 0.02	0.81 ± 0.01	0.76 ± 0.01	0.63 ± 0.01	0.77 ± 0.01	0.72 ± 0.04	0.56 ± 0.01	0.83 ± 0.00

Table 4: Fusion model performance across datasets and training data settings with test set performance reported using AUC, averaged over 3 runs, with standard deviation. Performance is compared to the best-performing ST adapter models (cf. Table 2 for all ST adapter results). Orange and teal coloring refer to a decrease and increase in absolute results, respectively, while gray coloring denotes similar performance. Using ASO testing, we determine almost stochastic dominant models ($\epsilon_{\min} < \tau$ with $\tau = 0.5$), marked in bold. While changes in performance are statistically significant, the absolute magnitude of performance increase and decrease remains within maximum +2% and -6%.

Task	Abbr.	Dataset Source	Adapter Source
(Morpho-)Syntactic			
Chunking	chunk	(Tjong Kim Sang and Buchholz, 2000)	AH/r-b-pf-conll2000
Dependency Relation Class.	deprel	(Nivre et al., 2017)	AH/r-b-pf-ud_deprel
Grammatical Error Detect.	ged	(Yannakoudakis et al., 2011)	AH/r-b-pf-fce_error_detection
Linguistic Acceptability	la	(Warstadt et al., 2019)	lingaccept/cola@ukp
Named Entity Recognition	ner	Link only[9]	AH/r-b-pf-mit_movie_trivia
Named Entity Recognition	ner	(Tjong Kim Sang and De Meulder, 2003)	AH/r-b-pf-conll2003
Named Entity Recognition	ner	(Derczynski et al., 2017)	AH/r-b-pf-wnut_17
Parsing	parse	(Nivre et al., 2017)	AH/r-b-pf-ud_en_ewt
Tagging	tag	(Tjong Kim Sang and De Meulder, 2003)	AH/r-b-pf-conll2003_pos
Tagging	tag	(Nivre et al., 2017)	AH/r-b-pf-ud_pos
Tagging	tag	(Abzianidze et al., 2017)	AH/r-b-pf-pmb_sem_tagging
Lexical Semantics			
Abstractness Prediction	abstr	(Brysbaert et al., 2014)	See our code repo
Emotion Analysis	emo	(Chatterjee et al., 2019)	AH/r-b-pf-emo
Sentiment Analysis	senti	(Maas et al., 2011)	AH/r-b-pf-imdb
Sentiment Analysis	senti	(Pang and Lee, 2005)	AH/r-b-pf-rotten_tomatoes
Sentiment Analysis	senti	(Socher et al., 2013)	sentiment/sst-2@ukp
Sentiment Analysis	senti	(Zhang et al., 2015)	AH/r-b-pf-yelp_polarity
Selectional Preference Pred.	sp	(Zhang et al., 2019)	See our code repo
Sentence-/Discourse-level Semantics			
Argument Mining	arg	(Stab et al., 2018)	argument/ukpsent@ukp
Commonsense Reasoning	csr	(Sap et al., 2019)	comsense/siqa@ukp
Commonsense Reasoning	csr	(Bhagavatula et al., 2020)	AH/r-b-pf-art
Commonsense Reasoning	csr	(Gordon et al., 2012)	AH/r-b-pf-copa
Commonsense Reasoning	csr	(Huang et al., 2019)	AH/r-b-pf-cosmos_qa
Commonsense Reasoning	csr	(Talmor et al., 2019)	AH/r-b-pf-commonsense_qa
Commonsense Reasoning	csr	(Zellers et al., 2019)	AH/r-b-uncased-pf-hellaswag
Commonsense Reasoning	csr	(Sakaguchi et al., 2021)	AH/r-b-pf-winogrande
Machine-Reading Compr.	mrc	(Rogers et al., 2020)	AH/r-b-pf-quail
Machine-Reading Compr.	mrc	(Khashabi et al., 2018)	AH/r-b-pf-multirc
Machine-Reading Compr.	mrc	(Lai et al., 2017)	AH/r-b-pf-race
Machine-Reading Compr.	mrc	(Zhang et al., 2018)	AH/r-b-pf-record
Natural Lanaguge Inf.	nli	(Williams et al., 2018)	nli/multinli@ukp
Natural Lanaguge Inf.	nli	(Dagan et al., 2006)	nli/rte@ukp
Natural Lanaguge Inf.	nli	(Nie et al., 2020)	AH/r-b-pf-anli_r3
Natural Lanaguge Inf.	nli	(de Marneffe et al., 2019)	nli/cb@ukp
Natural Lanaguge Inf.	nli	(Wang et al., 2019)	nli/qnli@ukp
Natural Lanaguge Inf.	nli	(Khot et al., 2018)	AH/r-b-pf-scitail
Natural Lanaguge Inf.	nli	(Marelli et al., 2014)	AH/r-b-pf-sick
Natural Lanaguge Inf.	nli	(Bowman et al., 2015)	AH/r-b-pf-snli
Natural Lanaguge Inf.	nli	(Zellers et al., 2019)	AH/r-b-pf-swag
Question Answering	qa	(Dua et al., 2019)	AH/r-b-pf-drop
Question Answering	qa	(Rajpurkar et al., 2016)	qa/squad1@ukp
Question Answering	qa	(Rajpurkar et al., 2018)	qa/squad2@ukp
Question Answering	qa	(Clark et al., 2019)	AH/r-b-pf-boolq
Question Answering	qa	(Abujabal et al., 2019)	AH/r-b-pf-comqa
Question Answering	qa	(Talmor and Berant, 2018)	AH/r-b-pf-cq
Question Answering	qa	(Saha et al., 2018)	AH/r-b-pf-duorc_s
Question Answering	qa	(Yang et al., 2018)	AH/r-b-pf-hotpotqa
Question Answering	qa	(Trischler et al., 2017)	AH/r-b-pf-newsqa
Question Answering	qa	(Tafjord et al., 2019)	AH/r-b-pf-quartz
Question Answering	qa	(Dasigi et al., 2019)	AH/r-b-pf-quoref
Question Answering	qa	(Welbl et al., 2018)	AH/r-b-pf-wikihop
Semantic Textual Similarity	sts	(Cer et al., 2017)	sts/sts-b@ukp
Semantic Textual Similarity	sts	(Dolan and Brockett, 2005)	AH/r-b-pf-mrpc
Semantic Textual Similarity	sts	Link only[10]	AH/r-b-pf-qqp

Table 5: Overview of tasks adapters. Categorization into *tasks* follows Adapterhub[11] sorting where possible. Task *Abbr.* refer to abbreviations as used in this paper. *Dataset source* denotes the dataset used to train an adapter with a reference to a paper or, where no paper could be found, a link to a website with a description. Adapter Source denotes the source where an existing adapter was harnessed from. For the sake of space, we abbreviate `AH/roberta-base` with `AH/r-b` which should be correspondingly expanded when searching for a given adapter. Please see `https://github.com/AnneroseEichel/Adapters-for-PP` for details on where to find our adapters.

Investigating Multi-Pivot Ensembling with Massively Multilingual Machine Translation Models

Alireza Mohammadshahi [*1,2] **Jannis Vamvas** [1] **Rico Sennrich** [1]
[1] University of Zurich [2] EPFL
alireza.mohammadshahi@epfl.ch
{vamvas,sennrich}@cl.uzh.ch

Abstract

Massively multilingual machine translation models allow for the translation of a large number of languages with a single model, but have limited performance on low- and very-low-resource translation directions. Pivoting via high-resource languages remains a strong strategy for low-resource directions, and in this paper we revisit ways of pivoting through multiple languages. Previous work has used a simple averaging of probability distributions from multiple paths, but we find that this performs worse than using a single pivot, and exacerbates the hallucination problem because the same hallucinations can be probable across different paths. We also propose MaxEns, a novel combination strategy that makes the output biased towards the most confident predictions, hypothesising that confident predictions are less prone to be hallucinations. We evaluate different strategies on the FLORES benchmark for 20 low-resource language directions, demonstrating that MaxEns improves translation quality for low-resource languages while reducing hallucination in translations, compared to both direct translation and an averaging approach. On average, multi-pivot strategies still lag behind using English as a single pivot language, raising the question of how to identify the best pivoting strategy for a given translation direction.[1]

1 Introduction

Early work on multilingual neural machine translation (NMT) has explored combining source segments in different source languages (Zoph and Knight, 2016; Firat et al., 2016a), an idea that is also compatible with pivoting through intermediate languages. For example, one could translate from Dutch to Ukrainian by first translating the Dutch source to English and Russian, and then making a combined prediction to Ukrainian. In the simplest case, this combination is achieved by predicting probability distributions for each source language and averaging these predictions in an ensemble-like manner (Firat et al., 2016a).

With massively multilingual NMT models (NLLB Team et al., 2022; Mohammadshahi et al., 2022a; Goyal et al., 2022; Wenzek et al., 2021; Zhang et al., 2020; Fan et al., 2021; Aharoni et al., 2019; Arivazhagan et al., 2019), one can in principle translate directly in any translation direction. While early models relied on zero-shot generalization for many directions, recent improvements include massive data collection efforts (Schwenk et al., 2021; El-Kishky et al., 2020) and synthetic data creation via back-translation (Edunov et al., 2018; Sennrich et al., 2016). However, these models still have low performance on many low-resource translation directions[2] and pivot-translation via high-resource languages remains a strong baseline. Fan et al. (2021) also investigate the combination of multiple translation paths, which they call *multi-source self-ensemble*, that slightly improves over the direct translation and a single pivot for zero-shot language pairs.

In this paper, we investigate this multi-source self-ensembling strategy more closely, with a focus on preventing completely defunct translations such as hallucinations. However, we find that simple averaging is sub-optimal and may increase the number of hallucinations in the output, a typical failure case in low-resource settings. We relate this to a recent finding that hallucinations are *sticky*, meaning that different models trained on the same data and architecture may produce similar hallucinations (Guerreiro et al., 2023a). We also find evidence of such stickiness when combining multiple translation paths, and propose a new ensembling strategy that, instead of averaging probabilities, picks the

[*] Work done while working at University of Zurich. Currently co-founder of Leeroo.

[1] The implementation is publicly available at https://github.com/ZurichNLP/MultiPivotNMT.

[2] SentencePiece BLEU of 63% translation directions in M2M-100 is lower than 12 (Mohammadshahi et al., 2022b).

Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, pages 169–180
June 20, 2024 ©2024 Association for Computational Linguistics

output with the maximum probability across different paths: **MaxEns**. This is partially inspired by the finding that model confidence is a good heuristic for avoiding hallucinations, which tend to be low-confidence predictions (Guerreiro et al., 2023b).

We perform experiments on the FLORES benchmark (Goyal et al., 2022) for 20 low-resource translation directions by using two massively multilingual NMT models, SMaLL100 (Mohammadshahi et al., 2022a) and M2M100 (Fan et al., 2021). Our results show that while the average ensemble outperforms the direct translation, it still underperforms using only English as a pivot, both in terms of spBLEU and the number of hallucinations. MaxEns performs significantly better than the averaging strategy for both translation performance and hallucination. Specially, MaxEns has competitive translation performance with English pivoting on average, but still lags behind it on the hallucination performance. To sum up, our contributions are:

- We explore why a naive multi-pivot strategy with massively multilingual models can underperform single-pivot translation. Then, we propose MaxEns, a more robust ensembling technique for multi-pivot translation with multilingual NMT models.
- We evaluate different ensembling strategies on 20 low-resource translation directions of FLORES benchmark, and demonstrate that multi-pivot ensembling still lags behind the English pivoting.

2 Related Work

Several approaches exploited different multi-pivoting methods to improve the performance of NMT models, specifically for low-resource language directions (Macháček et al., 2023; Dabre et al., 2021; Kim et al., 2019; Cheng et al., 2017; Firat et al., 2016b). Macháček et al. (2023) analyzed the robustness of multi-source NMT in transcription errors. Dabre et al. (2021) improved the performance of simultaneous NMT by translating the source language into pivot languages, then applying the multi-source translation method (Zoph and Knight, 2016). Firat et al. (2016b) proposed a novel zero-resource translation approach by exploiting the multi-way multilingual NMT model, introduced by Firat et al. (2016a), and improved the performance over traditional pivot-based translation (Wu and Wang, 2007; Utiyama and Isahara, 2007). Cheng et al. (2017) introduced the pivot-based NMT model by jointly training source-to-pivot and pivot-to-target directions. Currey and Heafield (2019) proposed an alternative method by applying a monolingual pivot-language data for zero-resource NMT via back-translation (Sennrich et al., 2016).

3 Ensembling Methods

When performing direct translation, the score of a translation Y given a source sequence X_{src} is computed as follows:

$$s(Y;X_{src}) = \sum_{i=1}^{|Y|} \log p(y_i|y_{<i},X_{src}) \quad (1)$$

where $p(y_i|y_{<i},X_{src})$ is the predicted probability of the i-th target token y_i given the previous tokens $y_{<i}$ and the source sequence X_{src}.

For multi-pivot ensembling, we select a set of pivot languages $M = \{\mu_1, \mu_2, ..., \mu_K\}$ and generate the corresponding pivot translations $X_M = \{X_{\mu_1}, X_{\mu_2}, ..., X_{\mu_K}\}$. The final translation is generated by ensembling predictions, conditioned on the individual pivot translations.

In the following, we describe two approaches for such an ensembling: the multilingual averaging method and our MaxEns approach.

Multilingual Average (MultiAvg). Inspired by Fan et al. (2021); Firat et al. (2016b), we average the predicted probabilities of a token y_i across all pivot languages:[3]

$$s(Y;X_M) = \sum_{i=1}^{|Y|} \log \frac{1}{|M|} \sum_{k=1}^{|M|} p(y_i|y_{<i},X_{\mu_k}). \quad (2)$$

where $|Y|$ and $|M|$ are the number of target tokens and pivots, respectively.

Maximum Ensemble (MaxEns). As novel combination strategy that biases the prediction towards the more confident pivot, we propose the following approach:

$$s(Y;X_M) = \sum_{i=1}^{|Y|} \max_{k=1}^{|M|} [\log p(y_i|y_{<i},X_{\mu_k})]. \quad (3)$$

[3]We tried both averaging probabilities and log-probabilities in preliminary experiments, and averaging probabilities worked better in terms of translation performance and hallucination.

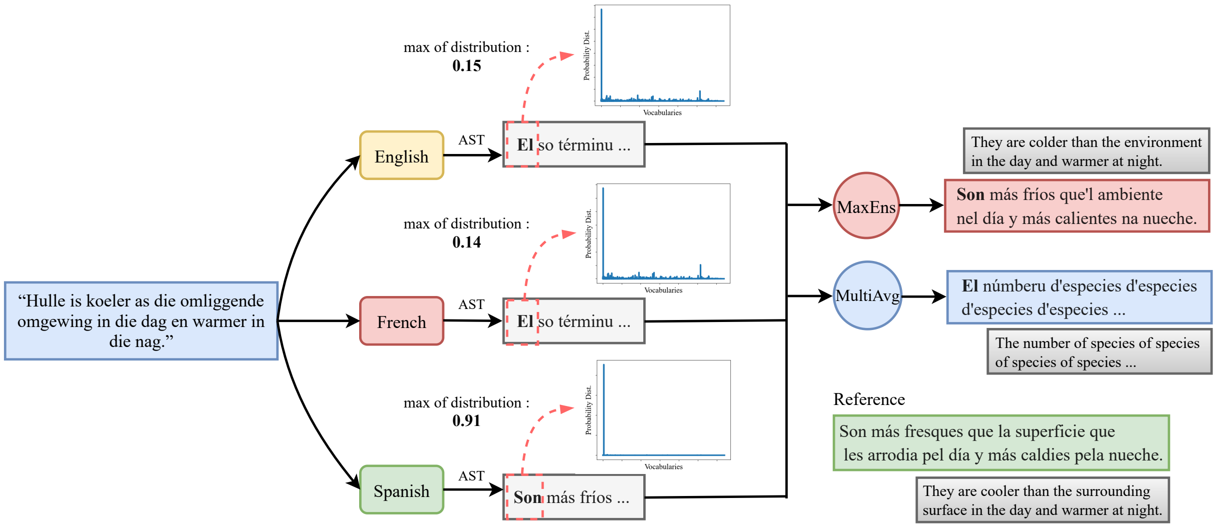


Figure 1: A sample translation of Afrikaans to Asturian by using SMaLL100. Translations of individual pivots are shown in the middle, output translations of MaxEns and MultiAvg on the right. MaxEns method eliminates the hallucination, as it follows the more confident pivot (here, Spanish). Glosses in English are presented within gray boxes.

where it chooses the maximum score between predictions of pivots for token y_i. Intuitively, MaxEns selects the most confident pivot language when generating token y_i.

4 Results and Discussion

4.1 Experimental Setup

Models. We used M2M100 and SMaLL100 as our massively multilingual NMT models. M2M100 is trained on large-scale multilingual corpora (Schwenk et al., 2021; El-Kishky et al., 2020) with a novel data mining procedure, that uses language similarities. We exploit M2M100 variant with 418M parameters. SMaLL100 (Mohammadshahi et al., 2022a) is a distilled version of M2M100 12B with 330M parameters. It has been trained with uniform sampling across all language pairs on nearly 6% of M2M100 pre-training dataset, and achieved competitive performance with M2M100 with 1.2B parameters.

Evaluation Setting. Inspired by Fan et al. (2021), we use the FLORES-101 benchmark (Goyal et al., 2022). It contains 3,001 sentences derived from English Wikipedia, and translated into 101 languages by human. We use the *devtest* subset for the evaluation. To better understand the effect of multilingual pivoting, we chose five low-resource (or very low) languages from different branches of Indo-European, including Germanic, Romance, Slavic, Indo-Aryan, and Iranian. These languages are Afrikaans, Asturian, Croatian, Urdu, and Pashto. We evaluate on all permutations of these languages, which results in 20 language pairs. As pivot languages, we use English, Spanish and French. English has the largest amount of bitext overall in the training data of M2M100, and Spanish and French have the largest amount of bitext with English (Fan et al., 2021).

Evaluation Metrics. spBLEU[4] is used to measure the translation performance (Goyal et al., 2022). For the hallucination measurement, we apply a coarse estimation method inspired by Lee et al. (2019); Müller and Sennrich (2021), counting the proportion of sentences with ChrF (Popović, 2015)[5] less than 20.[6] Additionally, we use top n-gram (TNG) (Guerreiro et al., 2023b; Raunak et al., 2022, 2021) for detecting oscillatory hallucinations.[7] We apply significance testing with $p = 0.05$.[8] Beam size 5 is used for inference.

4.2 Results & Discussion

Figure 1 illustrates an example of multi-pivot translation of Afrikaans to Asturian by using English, French, and Spanish as pivots. Translations via English and French pivots are hallucinations, while the translation via the Spanish pivot is more related to the reference translation. The output of MaxEns

[4]BLEU is computed after tokenization with SentencePiece with 256K tokens (Goyal et al., 2022).
[5]sacrebleu 2.3.1 (Post, 2018) with ChrF3 is used.
[6]Threshold based on manual inspection.
[7]We follow Guerreiro et al. (2023b) and use $n = 4$ and $t = 2$.
[8]Paired bootstrap resampling (Koehn, 2004) with sacrebleu.

Language Pairs	Direct	MultiAvg	MaxEns	EN Pivot
Afrikaans-Asturian	20.6	21.8	**22.5**	21.0
Afrikaans-Croatian	22.1	22.5	22.4	**22.8**
Afrikaans-Urdu	14.0	13.8	13.9	**14.4**
Afrikaans-Pashto	**5.9**	**5.8**	**6.0**	**6.0**
Asturian-Afrikaans	18.5	19.9	19.8	**21.0**
Asturian-Croatian	16.1	**20.4**	20.0	**20.4**
Asturian-Urdu	8.4	11.8	11.8	**12.6**
Asturian-Pashto	3.6	5.0	5.0	**5.4**
Croatian-Afrikaans	20.5	20.7	20.8	**21.2**
Croatian-Asturian	19.7	20.8	**21.6**	19.7
Croatian-Urdu	**13.5**	13.1	12.9	13.2
Croatian-Pashto	5.0	5.2	5.2	**5.6**
Urdu-Afrikaans	12.1	13.0	13.2	**13.8**
Urdu-Asturian	7.7	11.7	**12.8**	12.2
Urdu-Croatian	11.2	**12.0**	**11.9**	**12.2**
Urdu-Pashto	**4.7**	**4.4**	4.2	**4.6**
Pashto-Afrikaans	10.2	11.0	11.0	**11.4**
Pashto-Asturian	6.9	9.9	**10.9**	10.4
Pashto-Croatian	8.7	**9.9**	**10.0**	**9.8**
Pashto-Urdu	**10.0**	9.2	9.2	9.5
Average	12.0	13.1	**13.3**	**13.4**

Table 1: Average spBLEU (higher is better) of different pivoting methods for M2M100 and SMaLL100 on selected language pairs of FLORES-101. Best systems (not significantly outperformed by any other) in bold.

Language Pairs	Direct	MultiAvg	MaxEns	EN Pivot
Afrikaans-Asturian	5.9	6.4	4.0	4.3
Afrikaans-Croatian	1.7	2.2	2.2	1.9
Afrikaans-Urdu	11.3	13.9	13.4	11.1
Afrikaans-Pashto	49.5	54.1	53.2	49.8
Asturian-Afrikaans	8.8	6.2	7.8	2.1
Asturian-Croatian	19.5	5.3	8.2	3.5
Asturian-Urdu	40.7	23.4	22.9	18.0
Asturian-Pashto	66.7	62.2	62.2	55.9
Croatian-Afrikaans	1.1	1.5	1.2	1.3
Croatian-Asturian	6.1	6.3	3.8	5.0
Croatian-Urdu	13.2	16.5	16.3	12.7
Croatian-Pashto	54.7	58.4	56.6	53.1
Urdu-Afrikaans	9.8	9.2	9.4	4.9
Urdu-Asturian	31.5	19.9	14.7	12.5
Urdu-Croatian	14.1	15.4	15.5	11.8
Urdu-Pashto	56.8	66.0	66.8	60.1
Pashto-Afrikaans	9.1	10.5	10.0	6.6
Pashto-Asturian	26.9	21.2	15.7	15.6
Pashto-Croatian	18.4	18.5	18.7	16.7
Pashto-Urdu	24.7	33.2	33.3	28.9
Average	23.5	22.5	21.8	18.8

Table 2: Average percentage (100%) of hallucinations (chrF < 20; lower is better) of different pivoting methods for M2M100 and SMaLL100 on selected language pairs of FLORES-101.

approach is closer to the translation achieved by using Spanish as the pivot language, since the NMT model is more confident for this pivot (maximum of output probability distributions for the first token of English, French, and Spanish pivots are 0.15, 0.14, and 0.91, respectively). In contrast, the output of the MultiAvg method is a hallucination. Tables 1 and 2 show translation and hallucination performances on 20 language directions, respectively.[9] TNG scores for measuring oscillatory hallucinations are provided in Appendix B. MultiAvg approach achieves better translation performance and lower hallucination compared to the direct translation. However, MultiAvg method lags behind the English pivoting approach in terms of both translation quality (13.4 vs. 13.1) and the occurrence of hallucinations (18.8% vs. 22.5%).[10] Applying the MaxEns method instead for combining pivots tightens this gap, and leads to better translation and reduces the occurrence of hallucinations. Specifically, MaxEns reaches competitive translation quality with English pivoting on average (13.3 vs. 13.4), while still underperforming on hallucinations, as most of the parallel sentences of pre-training data for M2M100 and SMaLL100 are paired with English.

In general, the optimal strategy differs across translation directions, highlighting the potential for future research on determining the most effective translation strategy for each direction without depending on the development data for each.

5 Conclusion

We investigate more closely the multi-source self-ensembling method of Fan et al. (2021) for combining multiple translation paths to improve translations of low-resource (or very-low) language pairs. Specifically, this approach (named MultiAvg, here) averages the predictions of probability distributions of each source language in an ensemble-like manner. We evaluated it on 20 low-resource language pairs of FLORES-101 benchmark by using two massively multilingual NMT models, SMALL100 and M2M100. The MultiAvg method performs better than direct translation in terms of both translation quality and hallucinations, however it lags behind applying only English as pivot. Then, we proposed MaxEns method, a novel combination

[9] Average scores of M2M100 and SMaLL100 are shown in Table 1 and 2. Individual scores are provided in Appendix A.

[10] 4.5% vs. 7.3% based on the TNG metric, as shown in Table 7.

method that chooses the maximum of prediction probabilities of pivots for each designated target token. This approach results in a better translation quality compared to MultiAvg, while reducing hallucinations. On average, it achieves competitive performance with English pivoting with regard to the translation quality metric, but performs worse with regard to the hallucination metric. The most effective translation strategy varies depending on the translation direction, suggesting the need for future research to identify the optimal strategy for each direction independently of the specific development data. We hope our findings are a starting point for the broader integration of ensemble techniques within the context of massively multilingual NMT.

The insights of our experiments, specifically the stickiness of hallucinations with different inputs, have inspired our follow-up work on source-contrastive decoding (Sennrich et al., 2024), which has empirically shown to be an effective strategy to mitigate hallucinations. Future work could revisit multi-pivot ensembling in combination with source-contrastive decoding.

Limitations

We apply our method to two common massively multilingual NMT models, SMALL100 and M2M100; future work can extend our work to more recent large models e.g. NLLB200 (NLLB Team et al., 2022) and LLMs (Touvron et al., 2023; Workshop et al., 2022). We tested our approach on a subset of 20 low-resource language directions, future research can study the method for further language directions, including medium-resource language pairs.

Acknowledgement

This work was funded by the Swiss National Science Foundation (project MUTAMUR; no. 176727/213976).

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.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges.

Yong Cheng, Qian Yang, Yang Liu, 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, IJCAI-17*, pages 3974–3980.

Anna Currey and Kenneth Heafield. 2019. Zero-resource neural machine translation with monolingual pivot data. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 99–107, Hong Kong. Association for Computational Linguistics.

Raj Dabre, Aizhan Imankulova, Masahiro Kaneko, and Abhisek Chakrabarty. 2021. Simultaneous multi-pivot neural machine translation.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Ahmed El-Kishky, Vishrav Chaudhary, Francisco Guzmán, and Philipp Koehn. 2020. CCAligned: A massive collection of cross-lingual web-document pairs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5960–5969, Online. Association for Computational Linguistics.

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.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016a. 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.

Orhan Firat, Baskaran Sankaran, Yaser Al-onaizan, Fatos T. Yarman Vural, and Kyunghyun Cho. 2016b. Zero-resource translation with multi-lingual neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 268–277, Austin, Texas. Association for Computational Linguistics.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2022. The Flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

Nuno M. Guerreiro, Duarte M. Alves, Jonas Waldendorf, Barry Haddow, Alexandra Birch, Pierre Colombo, and André F. T. Martins. 2023a. Hallucinations in Large Multilingual Translation Models. *Transactions of the Association for Computational Linguistics*, 11:1500–1517.

Nuno M. Guerreiro, Elena Voita, and André Martins. 2023b. Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1059–1075, Dubrovnik, Croatia. Association for Computational Linguistics.

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.

Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. 2019. Hallucinations in neural machine translation.

Dominik Macháček, Peter Polak, Ondřej Bojar, and Raj Dabre. 2023. Robustness of multi-source MT to transcription errors. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3707–3723, Toronto, Canada. Association for Computational Linguistics.

Alireza Mohammadshahi, Vassilina Nikoulina, Alexandre Berard, Caroline Brun, James Henderson, and Laurent Besacier. 2022a. SMaLL-100: Introducing shallow multilingual machine translation model for low-resource languages. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8348–8359, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Alireza Mohammadshahi, Vassilina Nikoulina, Alexandre Berard, Caroline Brun, James Henderson, and Laurent Besacier. 2022b. What do compressed multilingual machine translation models forget? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4308–4329, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mathias Müller and Rico Sennrich. 2021. Understanding the properties of minimum Bayes risk decoding in 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 259–272, Online. Association for Computational Linguistics.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

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, Lisbon, Portugal. 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.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. 2021. The curious case of hallucinations in neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1172–1183, Online. Association for Computational Linguistics.

Vikas Raunak, Matt Post, and Arul Menezes. 2022. SALTED: A framework for SAlient long-tail translation error detection. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5163–5179, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. 2021. CCMatrix: Mining billions of high-quality parallel sentences on the web. 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 6490–6500, Online. Association for Computational Linguistics.

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, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Jannis Vamvas, and Alireza Mohammadshahi. 2024. Mitigating hallucinations and off-target machine translation with source-contrastive and language-contrastive decoding. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 21–33, St. Julian's, Malta. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Masao Utiyama and Hitoshi Isahara. 2007. A comparison of pivot methods for phrase-based statistical machine translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 484–491, Rochester, New York. Association for Computational Linguistics.

Guillaume Wenzek, Vishrav Chaudhary, Angela Fan, Sahir Gomez, Naman Goyal, Somya Jain, Douwe Kiela, Tristan Thrush, and Francisco Guzmán. 2021. Findings of the WMT 2021 shared task on large-scale multilingual machine translation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 89–99, Online. Association for Computational Linguistics.

BigScience Workshop, :, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, Dragomir Radev, Eduardo González Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady Elsahar, Hamza Benyamina, Hieu Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jörg Frohberg, Joseph Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro Von Werra, Leon Weber, Long Phan, Loubna Ben allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, María Grandury, Mario Šaško, Max Huang, Maximin Coavoux, Mayank Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad A. Jauhar, Mustafa Ghaleb, Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto Luis López, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, Shayne Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Davut Emre Taşar, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Fevry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiangru Tang, Zheng-Xin Yong, Zhiqing Sun, Shaked Brody, Yallow Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre François Lavallée, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure Ligozat, Arjun Subramonian, Aurélie Névéol, Charles Lovering, Dan Garrette, Deepak Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Jordan Clive, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, Shani Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdeněk Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ana Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Ajibade, Bharat Saxena, Carlos Muñoz Ferrandis, Daniel McDuff, Danish Contractor, David Lansky, Davis David, Douwe Kiela, Duong A. Nguyen, Edward Tan, Emi Baylor, Ezinwanne Ozoani, Fatima Mirza, Frankline Ononiwu, Habib Rezanejad, Hessie Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jesse Passmore, Josh Seltzer, Julio Bonis Sanz, Livia Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, Muhammed Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nour Fahmy, Olanrewaju Samuel, Ran An, Rasmus Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas Wang, Sourav Roy, Sylvain Viguier, Thanh Le, Tobi Oyebade, Trieu Le, Yoyo Yang, Zach Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Singh, Benjamin Beilharz, Bo Wang, Caio Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel León Periñán, Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrimann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Imane Bello, Ishani Dash, Jihyun Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthik Rangasai Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, Maria A Castillo, Marianna Nezhurina, Mario Sänger, Matthias Samwald, Michael Cullan, Michael Weinberg, Michiel De Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patrick Haller, Ramya Chandrasekhar, Renata Eisenberg, Robert Martin, Rodrigo Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aroonsiri, Srishti Kumar, Stefan Schweter, Sushil Bharati, Tanmay Laud, Théo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yash Shailesh Bajaj, Yash Venkatraman, Yifan Xu, Yingxin Xu, Yu Xu, Zhe Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. 2022. Bloom: A 176b-parameter open-access multilingual language model.

Hua Wu and Haifeng Wang. 2007. Pivot language approach for phrase-based statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 856–863, Prague, Czech Republic. Association for Computational Linguistics.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

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

Appendix A Individual Scores of M2M100 and SMaLL100 Models

A.A M2M100 Results

Language Pairs	Direct	MultiAvg	MaxEns	EN Pivot
Afrikaans-Asturian	19.3	20.2	**21.0**	20.2
Afrikaans-Croatian	**20.8**	**21.1**	**21.0**	**21.4**
Afrikaans-Urdu	14.0	13.6	13.8	**14.4**
Afrikaans-Pashto	**5.4**	**5.4**	**5.5**	**5.6**
Asturian-Afrikaans	14.2	16.5	16.1	**18.0**
Asturian-Croatian	11.1	**19.0**	18.3	**19.4**
Asturian-Urdu	6.3	11.4	11.4	**12.6**
Asturian-Pashto	2.4	4.4	4.5	**5.0**
Croatian-Afrikaans	17.6	**17.9**	**17.9**	**18.2**
Croatian-Asturian	18.8	19.5	**20.6**	19.1
Croatian-Urdu	**13.6**	13.3	13.1	**13.4**
Croatian-Pashto	4.4	4.9	4.9	**5.4**
Urdu-Afrikaans	9.0	9.8	10.0	**10.6**
Urdu-Asturian	7.1	9.9	**10.8**	**10.9**
Urdu-Croatian	8.9	**10.0**	**9.8**	**10.1**
Urdu-Pashto	**4.2**	3.8	3.6	**4.2**
Pashto-Afrikaans	8.3	**9.3**	8.9	**9.3**
Pashto-Asturian	7.8	9.6	**10.4**	9.7
Pashto-Croatian	8.0	**9.0**	**9.0**	8.5
Pashto-Urdu	**9.8**	8.9	8.9	9.0
Average	10.6	11.9	12.0	**12.2**

Table 3: spBLEU (higher is better) of different pivoting methods for M2M100 model on selected language pairs of FLORES-101 (Goyal et al., 2022) benchmark. Best systems (not significantly outperformed by any other) in bold.

Language Pairs	Direct	MultiAvg	MaxEns	EN Pivot
Afrikaans-Asturian	7.0	8.8	5.0	3.6
Afrikaans-Croatian	2.6	2.6	2.4	2.3
Afrikaans-Urdu	12.2	15.5	14.5	11.5
Afrikaans-Pashto	53.6	59.7	57.0	54.8
Asturian-Afrikaans	15.7	10.0	13.0	2.8
Asturian-Croatian	35.0	7.3	12.6	4.4
Asturian-Urdu	55.8	26.7	27.3	18.4
Asturian-Pashto	73.4	68.0	67.8	58.6
Croatian-Afrikaans	1.4	1.9	1.5	1.7
Croatian-Asturian	6.5	8.4	4.5	3.9
Croatian-Urdu	13.3	17.0	16.4	13.6
Croatian-Pashto	59.1	62.0	58.8	57.5
Urdu-Afrikaans	15.2	14.5	14.8	7.7
Urdu-Asturian	38.9	28.0	21.5	15.9
Urdu-Croatian	20.3	22.3	22.2	17.9
Urdu-Pashto	60.2	72.7	73.3	65.1
Pashto-Afrikaans	11.5	12.8	12.7	9.0
Pashto-Asturian	23.1	25.1	17.9	18.0
Pashto-Croatian	20.5	21.6	22.5	21.1
Pashto-Urdu	26.6	35.9	35.5	31.9
Average	27.6	26.0	25.0	21.0

Table 4: The percentage of hallucinations (chrF < 20; lower is better) of different pivoting methods for M2M100 model on selected language pairs of FLORES-101 (Goyal et al., 2022) benchmark.

A.B SMaLL100 Results

Language Pairs	Direct	MultiAvg	MaxEns	EN Pivot
Afrikaans-Asturian	22.0	23.4	**24.0**	21.7
Afrikaans-Croatian	23.5	**23.9**	**23.8**	**24.1**
Afrikaans-Urdu	13.9	14.0	14.0	**14.4**
Afrikaans-Pashto	**6.4**	6.1	**6.4**	**6.4**
Asturian-Afrikaans	22.8	**23.3**	**23.4**	**23.8**
Asturian-Croatian	21.1	**21.8**	**21.6**	**21.4**
Asturian-Urdu	10.5	12.1	12.2	**12.5**
Asturian-Pashto	4.8	**5.6**	**5.5**	**5.7**
Croatian-Afrikaans	23.4	23.4	23.7	**24.2**
Croatian-Asturian	20.6	**22.1**	**22.5**	20.2
Croatian-Urdu	**13.3**	12.8	12.6	**13.0**
Croatian-Pashto	**5.6**	**5.4**	**5.4**	**5.8**
Urdu-Afrikaans	15.1	16.1	16.3	**17.0**
Urdu-Asturian	8.3	13.4	**14.8**	13.4
Urdu-Croatian	13.4	14.0	13.9	14.3
Urdu-Pashto	**5.1**	**5.0**	**4.8**	**5.0**
Pashto-Afrikaans	12.0	12.7	12.9	**13.5**
Pashto-Asturian	6.0	10.2	**11.3**	**11.1**
Pashto-Croatian	9.4	**10.8**	**11.0**	**11.0**
Pashto-Urdu	**10.2**	9.5	9.4	9.9
Average	13.4	14.3	**14.5**	**14.4**

Table 5: spBLEU (higher is better) of different pivoting methods for SMaLL100 model on selected language pairs of FLORES-101 (Goyal et al., 2022) benchmark. Best systems (not significantly outperformed by any other) in bold.

Language Pairs	Direct	MultiAvg	MaxEns	EN Pivot
Afrikaans-Asturian	4.7	4.1	2.9	5.0
Afrikaans-Croatian	0.9	1.9	2.1	1.5
Afrikaans-Urdu	10.4	12.3	12.2	10.8
Afrikaans-Pashto	45.4	48.4	49.4	44.8
Asturian-Afrikaans	1.9	2.5	2.6	1.5
Asturian-Croatian	4.1	3.3	3.9	2.7
Asturian-Urdu	25.6	20.1	18.5	17.5
Asturian-Pashto	59.9	56.5	56.7	53.3
Croatian-Afrikaans	0.8	1.1	0.9	0.9
Croatian-Asturian	5.6	4.2	3.1	6.1
Croatian-Urdu	13.2	16.0	16.1	11.8
Croatian-Pashto	50.3	54.7	54.5	48.7
Urdu-Afrikaans	4.4	4.0	4.0	2.1
Urdu-Asturian	24.1	11.8	7.9	9.0
Urdu-Croatian	7.9	8.5	8.8	5.7
Urdu-Pashto	53.4	59.3	60.3	55.2
Pashto-Afrikaans	6.7	8.3	7.3	4.2
Pashto-Asturian	30.7	17.3	13.6	13.2
Pashto-Croatian	16.4	15.4	14.9	12.4
Pashto-Urdu	22.7	30.5	31.1	25.9
Average	19.5	19.0	18.5	16.6

Table 6: The percentage of hallucinations (chrF < 20; lower is better) of different pivoting methods for SMaLL100 model on selected language pairs of FLORES-101 (Goyal et al., 2022) benchmark.

Appendix B Results of oscillatory hallucinations based on TNG metric

Language Pairs	Direct	MultiAvg	MaxEns	EN Pivot
Afrikaans-Asturian	2.2	2.1	0.7	1.3
Afrikaans-Croatian	0.3	0.3	0.2	0.2
Afrikaans-Urdu	1.2	1.5	1.1	0.6
Afrikaans-Pashto	6.7	6.7	5.1	4.7
Asturian-Afrikaans	5.1	3.0	3.9	0.2
Asturian-Croatian	11.9	1.5	3.0	0.4
Asturian-Urdu	15.7	4.1	4.3	0.7
Asturian-Pashto	20.9	12.8	11.6	5.0
Croatian-Afrikaans	0.1	0.2	0.2	0.0
Croatian-Asturian	2.2	2.0	1.0	1.2
Croatian-Urdu	1.5	1.3	1.1	0.5
Croatian-Pashto	7.2	6.4	4.9	4.3
Urdu-Afrikaans	8.2	3.3	3.3	0.9
Urdu-Asturian	15.8	6.2	4.0	2.4
Urdu-Croatian	2.5	2.6	2.8	0.8
Urdu-Pashto	9.3	11.7	9.5	4.2
Pashto-Afrikaans	2.7	3.6	3.5	0.8
Pashto-Asturian	16.0	6.2	4.1	2.0
Pashto-Croatian	3.6	2.9	3.0	1.0
Pashto-Urdu	2.5	4.8	4.3	1.6
Average	7.3	4.5	4.0	1.9

Table 7: Average percentage (100%) of hallucinations (TNG metric; lower is better) of different pivoting methods for M2M100 and SMaLL100 on selected language pairs of FLORES-101.

Language Pairs	Direct	MultiAvg	MaxEns	EN Pivot
Afrikaans-Asturian	1.8	2.6	1.1	0.5
Afrikaans-Croatian	0.5	0.49	0.2	0.3
Afrikaans-Urdu	1.9	2.7	1.6	0.9
Afrikaans-Pashto	10.4	11.6	9.1	8.0
Asturian-Afrikaans	9.8	5.6	6.8	0.2
Asturian-Croatian	22.6	2.5	5.2	0.4
Asturian-Urdu	25.3	7.6	7.7	1.2
Asturian-Pashto	36.8	23.6	21.3	9.2
Croatian-Afrikaans	0.1	0.2	0.2	0.0
Croatian-Asturian	2.3	2.7	1.1	0.2
Croatian-Urdu	2.2	2.1	1.7	0.7
Croatian-Pashto	11.8	11.1	8.5	7.5
Urdu-Afrikaans	15.2	6.1	6.1	1.5
Urdu-Asturian	18.2	9.0	6.8	2.1
Urdu-Croatian	4.1	4.8	4.9	1.2
Urdu-Pashto	13.3	21.0	16.8	7.4
Pashto-Afrikaans	4.2	4.3	4.7	1.1
Pashto-Asturian	12.7	6.9	5.1	1.2
Pashto-Croatian	4.2	3.7	3.7	1.6
Pashto-Urdu	3.3	7.5	6.2	2.8
Average	10.9	7.6	6.7	2.8

Table 8: The percentage of hallucinations (TNG metric; lower is better) of different pivoting methods for M2M100 model on selected language pairs of FLORES-101 (Goyal et al., 2022) benchmark.

Language Pairs	Direct	MultiAvg	MaxEns	EN Pivot
Afrikaans-Asturian	2.5	1.6	0.2	2.1
Afrikaans-Croatian	0.0	0.1	0.2	0.1
Afrikaans-Urdu	0.4	0.3	0.6	0.2
Afrikaans-Pashto	2.9	1.8	1.2	1.3
Asturian-Afrikaans	0.4	0.5	0.9	0.2
Asturian-Croatian	1.2	0.4	0.7	0.4
Asturian-Urdu	6.1	0.7	0.9	0.2
Asturian-Pashto	5.0	2.0	1.9	0.9
Croatian-Afrikaans	0.0	0.1	0.1	0.0
Croatian-Asturian	2.1	1.3	0.8	2.2
Croatian-Urdu	0.8	0.4	0.4	0.3
Croatian-Pashto	2.6	1.7	1.3	1.2
Urdu-Afrikaans	1.1	0.6	0.4	0.3
Urdu-Asturian	13.4	3.3	1.2	2.6
Urdu-Croatian	0.8	0.4	0.7	0.4
Urdu-Pashto	5.3	2.3	2.1	1.0
Pashto-Afrikaans	1.2	2.9	2.2	0.4
Pashto-Asturian	19.4	5.4	3.1	2.7
Pashto-Croatian	3.0	2.1	2.3	0.3
Pashto-Urdu	1.6	2.2	2.4	0.4
Average	3.6	1.5	1.2	0.9

Table 9: The percentage of hallucinations (TNG metric; lower is better) of different pivoting methods for SMALL100 model on selected language pairs of FLORES-101 (Goyal et al., 2022) benchmark.

Author Index