ICNLSP 2023

Proceedings of the 6th International Conference on Natural Language and Speech Processing (ICNLSP-2023)

16–17 December, 2023 (virtual)

Order copies of this and other ACL proceedings from:

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

ISBN 979-8-89176-065-3

https://www.icnlsp.org

Introduction

Welcome to the proceedings of the 6th International Conference on Natural Language and Speech Processing!

This volume presents a vibrant tapestry of cutting-edge research in natural language processing, highlighting advancements in a diverse range of areas. It addresses many NLP aspects as bridging the language divide, expressive and robust communication, building and leveraging resources, and unifying theory and practice. Research works dealing with these topics have been presented at ICNLSP 2023.

Thirty seven (37) papers have been accepted by the program committee members that helped us a lot with their insightful comments. All papers have been presented orally, that is why the program was quite long and rich. The technical program included 05 oral sessions, namely: Classification and clustering, Deep learning and transformers, Analysis, summarization, and numerical representation, Speech and phonetics, and Dataset.

This year, we were honoured by the participation of two distinguished scholars: Prof. Dr. Alexander Waibel from Carnegie Mellon University (USA) and Karlsruhe Institute of Technology (Germany) and Dr. Najim Dehak from Johns Hopkins University (USA). Professor Alexander Waibel gave the first talk entitled “Transcending Communication Barriers: From Machine Translation to Language Transparence”. During his talk, Prof. Alex discussed the latest advances and activities to transcend these barriers. The second talk, entitled “Biosignal-based Digital Biomarkers for Aging” was given by Dr. Najim Dehak, in which he described several tools to detect, assess, and monitor the functional and cognitive decline of elderly adults. Both talks were very interesting.

This volume reflects the richness and diversity of the NLP community itself. Contributions from researchers across the globe explore a wide range of languages, domains, and methodologies. This tapestry of research highlights the collaborative spirit and boundless potential of NLP to revolutionize the way we understand, interact with, and create language.

We hope readers enjoy reading the content of the 6^{th} ICNLSP proceedings. We would like also to invite them to check the proceedings of the past versions of ICNLSP.

Mourad Abbas

Organizers:

General Chair: Dr. Mourad Abbas

Chair: Dr. Abed Alhakim Freihat

Program Chair: Dr. Mourad Abbas

Program Committee:

Ahmed Abdelali, SDAIA, Saudi Arabia.
Ahmed Ali, QCRI, Qatar.
Hend Al-Khalifa, King Saud University, Saudi Arabia.
Muhammad Al-Qurishi, Elm, Saudi Arabia.
Mehmet Fatih Amasyalı, Yildiz Technical University, Turkey.
Yuan An, Drexel University, USA.
Fayssal Bouarourou, University of Strasbourg, France.
Hadda Cherroun, Amar Telidji University, Algeria.
Gérard Chollet, CNRS, France.
Dirk Van Compernolle, KU Leuven, Belgium.
Najim Dehak, Johns Hopkins University, USA.
Ashraf Elnagar, University of Sharjah, UAE.
Abed Alhakim Freihat, University of Trento, Italy.
Munir Georges, Technische Hochschule Ingolstadt, Germany.
Ahmed Guessoum, USTHB, Algeria.
Kais Haddar, Faculty of Sciences of Sfax, Tunisia.
Valia Kordoni, Humboldt University, Germany.
Saurabh Kulshreshtha, University of Massachusetts Lowell, USA.
Eric Laporte, Gustave Eiffel University, France.
Mohamed Lichouri, USTHB, Algeria.
Mohammed Mediani, University of Ahmed Draia, Algeria.
Pascal Perrier, Université Grenoble Alpes, France.
Aishwarya Reganti, Amazon Alexa AI.
Hassan Satori, Sidi Mohammed Ben Abbdallah University, Morocco.
Tim Schlippe, IU International University of Applied Sciences, Germany.
Nasredine Semmar, CEA, France.
Deven Shah, Microsoft, USA.
R. V. Swaminathan, Amazon, USA.
Irina Temnikova, Big Data for Smart Society Institute, Bulgaria.
Jan Trmal, Johns Hopkins UniIU International University of Applied Sciencesversity, USA.
Fayçal Ykhlef, CDTA, Algeria.
Wajdi Zaghouani, Hamad Bin Khalifa University, Qatar.

Additional reviewers:

Arpit Sood, Meta Inc., USA.
Guanqun Yang, Stevens Institute of Technology, USA.
Abdelouahab Moussaoui, Ferhat Abbas University, Algeria.
Slimane Oulad Naoui, University of Ghardaia, Algeria.

Invited Speakers:

Prof. Dr. Alex Waibel, Carnegie Mellon University, USA.
Prof. Najim Dehak, Johns Hopkins University, USA.

Invited Talks

Transcending Communication Barriers: From Machine Translation to Language Transparence

Prof. Dr. Alex Waibel, Carnegie Mellon University, USA

As we marvel at impressive advances in Artificial Intelligence in recent years, we may wonder whether the problem of language translation and language barriers has been solved. Aside from remaining technical issues, it is important to note that translation is only one (even though important) step toward making people on the planet understand each other: Our thoughts are expressed in many ways: speech, text, video, handwriting, road signs, facial expressions, voice, lip movement, emotion, gesture, mannerisms and more… For frictionless communication, the way technology is deployed in different settings is just is as important a consideration as the performance of the technology itself and they come with profound consequences on the technical design and requirements. To make language barriers fade into the background, we need language transparence, not only translation: multimodal, immersive, cross lingual, culturally aware, proactive communication and dubbing tools that interpret the communicative intent and transcend barriers between us. In this talk, I will review major milestones on our journey and discuss our latest advances and activities toward this goal.

Biosignal-based Digital Biomarkers for Aging

Dr. Najim Dehak, Johns Hopkins University, USA

Currently, there are more Americans aged 65 and older (over 49 million) than at any other time in history, according to the US Census Bureau. A significant increase in individuals with severe chronic conditions will have profound social and economic effects on society. Three aspects describe the human aging process: functional (motor system), cognitive, and behavior (social and psychological stressors). In this talk, we will describe several tools to detect, assess, and monitor the functional and cognitive decline of elderly adults. Those tools named biomarkers are based on multimodal biosignals such as speech, handwriting, and eye movement. In addition, we will describe our current work on emotion recognition from speech that can be used to assess social and psychological stressors.

Table of Contents

Classification of Human- and AI-Generated Texts for English, French, German, and Spanish

Kristina Schaaff and **Tim Schlippe** and **Lorenz Mindner**
IU International University of Applied Sciences, Germany.
kristina.schaaff@iu.org; tim.schlippe@iu.org

Abstract

In this paper we analyze features to classify *human-* and *AI-generated* text for English, French, German and Spanish and compare them across languages. We investigate two scenarios: (1) The detection of text generated by AI from scratch, and (2) the detection of text rephrased by AI. For training and testing the classifiers in this multilingual setting, we created a new text corpus covering 10 topics for each language. For the detection of *AI-generated* text, the combination of all proposed features performs best, indicating that our features are portable to other related languages: The F1-scores are close with 99% for Spanish, 98% for English, 97% for German and 95% for French. For the detection of *AI-rephrased* text, the systems with all features outperform systems with other features in many cases, but using only document features performs best for German (72%) and Spanish (86%) and only text vector features leads to best results for English (78%).

1 Introduction

In recent years, chatbots have gained popularity and are now widely used in everyday life (Pelau et al., 2021). These systems are designed to simulate *human*-like conversations and provide assistance, information, and emotional support (Dibitonto et al., 2018; Arteaga et al., 2019; Falala-Séchet et al., 2019; Adiwardana et al., 2020). OpenAI's ChatGPT has emerged as one of the most commonly used tool for text generation (Taecharungroj, 2023). Within a short span of only five days after its release, over one million users registered (Taecharungroj, 2023). The application scenarios are manifold, ranging from children seeking help with their homework to individuals seeking medical advice or companionship.

As the use of chatbots like ChatGPT becomes more prevalent in our daily lives, it is important to differentiate between *human-generated* and *AI-generated* text. As AI algorithms improve, detecting *AI-generated* content accurately becomes increasingly challenging, posing issues such as plagiarism, fake news generation, and spamming. Thus, tools that can differentiate between *human-* and *AI-generated* content are crucial.

In Mindner et al. (2023), we explored a large number of innovative features such as text objectivity, list lookup features, and error-based features for the detection of English (*EN*) text generated by ChatGPT. However, in the current study, we extended this research to Spanish (*ES*), German (*DE*), and French (*FR*). We selected these languages, as these are amongst the most frequently used languages in the world (Ethnologue, 2023).

Consequently, our contributions are as follows:

- We proved, that the features we investigated in Mindner et al. (2023) can be successfully ported to other languages.
- We extended our *Human-AI-Generated Text Corpus*[1] with *FR*, *DE* and *ES* articles which cover 10 topics, providing a benchmark corpus for the detection of *AI-generated texts* in *EN*, *FR*, *DE* and *ES*.
- Our best systems significantly outperform the state-of-the-art system for the detection of *AI-generated* text ZeroGPT.

2 Related Work

In the this section, we will describe the related work concerning ChatGPT and the classification of *human-* and *AI-generated* texts.

2.1 ChatGPT

Since its release by OpenAI in late 2022, ChatGPT has revolutionized the field of AI (Mesko, 2023) and several other generative AIs such as Google's Bard[2] or Llama[3] (Touvron et al., 2023) have been

[1] https://github.com/LorenzM97/human-AI-generatedTextCorpus

[2] https://bard.google.com

[3] https://ai.meta.com/llama

released. Those tools are capable of generating text in response to user queries across a wide range of domains. Its successful implementation has been demonstrated in areas like education (Baidoo-Anu and Owusu Ansah, 2023), medicine (Jeblick et al., 2022), and language translation (Jiao et al., 2023). ChatGPT is built on the Generative Pre-trained Transformers (GPT) language model and undergoes fine-tuning using reinforcement learning with human feedback. This approach allows ChatGPT to grasp the meaning and intention behind user prompts, enabling it to provide relevant and helpful responses. During the training process, a substantial amount of text data is incorporated to ensure the safety and accuracy of the generated text. While the quantity of training data has not been published, we know that the previous GPT-3 model, which is substantially larger than other language models such as BERT (Kenton and Toutanova, 2019), RoBERTa (Liu et al., 2019), and T5 (Roberts et al., 2019), was trained with 175 billion parameters and 499 billion crawled text tokens (Brown et al., 2020). Through extensive training on a diverse dataset, ChatGPT has acquired a sophisticated understanding of human language, allowing it to generate text that closely resembles that written by humans (Mitrović et al., 2023).

2.2 Detecting *Human-* and *AI-Generated* Texts

Commercial tools and plagiarism apps, such as GPTZero (Shrivastava, 2023), ZeroGPT[4], AI Content Detector[5], and GPT-2 Output Detector[6] (Mitchell et al., 2023), have been developed to identify *AI-generated* text. Furthermore, researchers are working on developing new corpora for this task and finding out which features and classifiers improve classification accuracy: For example, (Yu et al., 2023) present a corpus of *human-* and *AI-generated* abstracts to investigate commercial and non-commercial systems—but only for *EN*. Recent studies have explored various approaches to detect *AI-generated* text, including XGBoost (Shijaku and Canhasi, 2023), decision trees (Zaitsu and Jin, 2023), and transformer-based models (Mitrović et al., 2023; Guo et al., 2023): Mitrović et al. (2023) evaluated characteristics of *AI-generated* text from *EN* customer reviews and built a transformer-based classifier that achieved 79%. Zaitsu and Jin (2023) achieved 100% accuracy in the detection of Japanese texts with decision trees combining stylometric features for Japanese such as bigrams, comma position, and function word rates. Guo et al. (2023) evaluated the characteristics of *human-generated* and *AI-generated* answers to questions in *EN* and Chinese. They fine-tuned a RoBERTa model on their texts and achieved 98.8% F1-score on the *EN* answers and 96.4% F1-score on the Chinese answers. Shijaku and Canhasi (2023) addressed the detection of generated essays written in *EN* and proposed an XGBoost model that achieved 98% accuracy using features generated by TF-IDF and a set of handcrafted features. Soni and Wade (2023) analyzed *human-* and *AI-generated* text summarization and achieved 90% accuracy using DistilBERT[7] (Sanh et al., 2019). Mindner et al. (2023) explored features to detect *AI-generated* and *-rephrased* text for *EN*. They report an F1-score of 96% for *AI-generated* text and 78% for *AI-rephrased* text on their text corpus which contains different topics. These F1-scores were even achieved when the AI was instructed to create the text in a way that a human would not recognize that it was generated by an AI.

To the best of our knowledge, we are the first to explore a large set of features and state-of-the-art classifiers across multiple languages with XGBoost, Random Forrest and MLP. We compare our results with two popular state-of-the-art tools that detect texts generated by AI: GPTZero and ZeroGPT. GPTZero is used by over 1 million people (Shrivastava, 2023), but its results are only reliable for *EN* texts. Consequently, we also used ZeroGPT for comparison which is able to deal with other languages. As there is currently no text corpus available, which contains *human-* and *AI-generated* texts in multiple languages, we extended our *Human-AI-Generated Text Corpus* to cover *EN*, *FR*, *DE* and *ES*.

3 Our Human-AI-Generated Text Corpus

As mentioned in the previous section, we extended our *Human-AI-Generated Text Corpus* (Mindner et al., 2023) to cover *EN*, *FR*, *DE*, and *ES*. In total, for each language we used 100 *human-generated*, 100 *AI-generated*, and 100 *AI-rephrased* articles for our multilingual analysis which contain the following 10 topics: $biology$, $chemistry$, $geography$, $history$, IT, $music$, $politics$, $religion$, $sports$, and $visualarts$.

[4] https://www.zerogpt.com

[5] https://copyleaks.com/ai-content-detector

[6] https://openai-openai-detector–mqlck.hf.space

[7] https://huggingface.co/docs/transformers/model_doc/distilbert

	Human			AI-generated			AI-rephrased		
Language	**P**	**S**	**W**	**P**	**S**	**W**	**P**	**S**	**W**
EN	415	1.7k	38.3k	555	1.4k	27.6k	255	1.1k	24.6k
FR	415	1.2k	31.0k	524	1.3k	26.5k	157	0.8k	18.7k
DE	335	1.2k	20.5k	529	1.4k	22.9k	256	1.0k	16.4k
ES	450	1.4k	38.0k	514	1.2k	26.8k	190	0.8k	18.9k

Table 1: *AI-Generated/Rephrased* Text
(P = #paragraphs, S = #sentences, W = #words).

The characteristics of our *Human-AI-Generated Text Corpus* for the respective languages are summarized in Table 1: *EN* consistently has the highest counts across all categories and types of text. On the other hand, the counts for *FR*, *DE*, and *ES* vary substantially depending on whether the text was *human-generated*, *AI-generated*, or *AI-rephrased*. This illustrates how languages differ in the expression of information. The prompts which we used to receive the *AI-generated* and *AI-rephrased* texts are listed in Table 2.

Lang.	**Prompt**
Text Generation	
EN	Generate a text on the following topic: <topic>
FR	Rédigez un texte sur le thème suivant: <topic>
DE	Erstelle einen Text zum folgenden Thema: <topic>
ES	Genera un texto sobre el siguiente tema: <topic>
Text Rephrasing	
EN	Rephrase the following text: <topic>
FR	Reformulez le texte suivant: <topic>
DE	Formuliere den folgenden Text um: <topic>
ES	Reformule el siguiente texto: <topic>

Table 2: Prompts used for Generation and Rephrasing

4 Our Features for the Classification of *Human-* and *AI-Generated* Texts

As shown in Table 3, we analyzed 37 features for their suitability to discriminate between *human-* and *AI-generated* text. More details of the features are given in Mindner et al. (2023).

4.1 Perplexity-Based Features

Perplexity is a measure of how well a language model is able to predict a sequence of words. The lower the perplexity, the better a language model will perform to predict the next word in a sequence. As *AI-generated* texts are usually based on statistical patterns and rules, they tend to be more repetitive and therefore have a lower perplexity than human generated texts. The *perplexity-based* features in our study are based on the findings by Mindner et al. (2023); Gehrmann et al. (2019); Mitrović et al. (2023); Guo et al. (2023).

For sentence tokenization, we use the Natural Language Toolkit (NLTK)[8]. Perplexity is calculated using *evaluate package*[9] and GPT-2 using the respective models for *EN*[10], *FR*[11], *DE*[12], and *ES*[13].

4.2 Semantic Features

In our study, *semantic* features refer to the properties of words or phrases used to represent their meanings. Previous studies successfully used these features for the differentiation between *human-* and *AI-generated* texts (Mitrović et al., 2023; Guo et al., 2023; Mindner et al., 2023).

Again, we use different Python packages for the respective languages: TextBlob's sentiment analysis for English[14], `textblob-fr`[15] for French, and `textblob-de`[16] for German. Due to the absence of a package that computes both, polarity and subjectivity, for *ES* texts were translated these texts into *EN* using Googletrans[17], despite potential information loss, because of its high BLEU score and proficiency in *ES-EN* translation.

4.3 List Lookup Features

With our *ListLookup* features, we analyze information about the word or character class, e.g., whether it is a stop word or a special character. These features have previously been used for this task by Mindner et al. (2023); Shijaku and Canhasi (2023); Kumarage et al. (2023). For every language, we used ChatGPT to generate a list of all discourse markers as well as the personal pronouns. These lists were additionally evaluated by language experts. To count stop words, we use NLTK for the respective languages.

[8] https://www.nltk.org
[9] https://github.com/huggingface/evaluate
[10] https://huggingface.co/gpt2
[11] https://huggingface.co/dbddv01/gpt2-french-small
[12] https://huggingface.co/dbmdz/german-gpt2
[13] https://huggingface.co/DeepESP/gpt2-spanish
[14] https://textblob.readthedocs.io/en/dev/quickstart.html
[15] https://github.com/sloria/textblob-fr
[16] https://textblob-de.readthedocs.io/en/latest/api_reference.html#module-textblob_de.sentiments
[17] https://github.com/ssut/py-googletrans

Category	**Feature**	**Description**
Perplexity	PPL_{mean}	mean PPL
	PPL_{max}	maximum PPL
Semantic	$sentiment_{polarity}$	degree of positivity/negativity [-1,+1]
	$sentiment_{subjectivity}$	degree of subjectivity [0,+1]
ListLookup	$stopWord_{count}$	number of stop words
	$discourseMarker_{count}$	number of discourse markers
	$titleRepetition_{count}$	absolute repetitions of title
	$titleRepetition_{relative}$	relative repetitions of title
	$personalPronoun_{count}$	absolute number of personal pronouns
	$personalPronoun_{relative}$	relative number of personal pronouns
Document	$wordsPerParagraph_{mean}$	mean number of words per paragraph
	$wordsPerParagraph_{stdev}$	stdev of $wordsPerParagraph$
	$sentencesPerParagraph_{mean}$	mean number of sentences per paragraph
	$sentencesPerParagraph_{stdev}$	stdev of $sentencesPerParagraph$
	$wordsPerSentence_{mean}$	mean number of words per sentence
	$wordsPerSentence_{stdev}$	stdev of $wordsPerSentence$
	$uniqWordsPerSentence_{mean}$	mean number of unique words per sentence
	$uniqWordsPerSentence_{stdev}$	stdev of $uniqWordsPerSentence$
	$words_{count}$	number of running words
	$uniqWords_{count}$	number of unique words
	$uniqWords_{relative}$	relative number of unique words
	$paragraph_{count}$	number of paragraphs
	$sentence_{count}$	number of sentences
	$punctuation_{count}$	number of punctuation marks
	$quotation_{count}$	number of quotation marks
	$character_{count}$	number of characters
	$uppercaseWords_{relative}$	relative number of words in uppercase
	$POSPerSentence_{mean}$	mean number of unique POS-tags/sentence
	$specialChar_{count}$	number of special characters
ErrorBased	$grammarError_{count}$	number of spelling/grammar errors
	$multiBlank_{count}$	number of multiple blanks
Readability	$fleschReadingEase$	Flesch Reading Ease score [0-100]
	$fleschKincaidGradeLevel$	Readability as U.S. grade level [0-100]
AIFeedback	$AIFeedback$	Ask AI if text was generated by AI
TextVector	*TF-IDF*	500-dim TF-IDF vector of 1-/2-grams
	Sentence-BERT	mean Sentence-BERT vector
	Sentence-BERT-dist	mean distance of Sentence-BERT vectors

Table 3: Summary of our Features for the Classification of Generated Texts.

4.4 Document Features

Our *document* features are related to the content and structure of a document such as word frequencies, syntactic structures, and corpus statistics. These features have been successfully used by (Kumarage et al., 2023; Shijaku and Canhasi, 2023; Guo et al., 2023; Mitrović et al., 2023; Zaitsu and Jin, 2023; Mindner et al., 2023). To calculate *sentence-* and *word-related* features, the text is first divided into sentences and words using NLTK's `sent_tokenize` and `word_tokenize` functions. For the features related to Part-of-speech (POS) in *EN* texts, we use the NLTK function `pos_tag`. As NLTK lacks POS tags for the other three languages, we use spaCy NLP library[18]. For POS tags in *DE* texts, we use `de_core_news_sm`[19], for *FR* texts, we use `fr_core_news_sm`[20], and for *ES* texts, `es_core_news_sm`[21].

4.5 Error Based Features

This feature category introduced in Mindner et al. (2023) is based on errors in the text such as grammar and spelling mistakes.

To count multiple blanks, we used regular expressions. Grammar and spelling errors are detected using the open-source tool *LanguageTool*[22] which allows it to detect grammar errors in multiple languages. For the detection of *DE* errors, the built-in class `LanguageToolPublicAPI(de-DE)` for querying the tool's public servers is used. For the other languages, the tool's remote server is applied using the function `Language-Tool(language)`.

[18] https://github.com/explosion/spaCy

[19] https://spacy.io/models/de#de_core_news_sm

[20] https://spacy.io/models/fr#fr_core_news_sm

[21] https://spacy.io/models/es#es_core_news_sm

[22] https://github.com/jxmorris12/language_tool_python

4.6 Readability Features

Readability features assess the readability level of texts as in Mindner et al. (2023); Shijaku and Canhasi (2023); Flesch (1948); Kincaid et al. (1975).

To derive Flesch Reading Ease and Flesch-Kincaid Grade Level we use *Textstat*[23]. This Python library provides functions to calculate text statistics such as grade level, complexity, and readability. Textstat supports calculating Flesch Reading Ease, and Flesch-Kincaid Grade Level for *EN*, *FR*, *DE*, and *ES* texts. However, it is important to note that these measures were originally developed for the specific structure of words, sentences, and syllables of *EN*. Therefore, when applying these measures to texts in *FR*, *DE*, and *ES*, the results may not be as representative as those for *EN*.

4.7 AI Feedback Features

Our *AI Feeback* features reflect, how an AI categorizes the text (Mindner et al., 2023). For this purpose, we use ChatGPT with the prompts in Table 4.

Lang.	Prompt
EN	Was the following text generated by ChatGPT?
FR	Le texte suivant a-t-il été généré par ChatGPT?
DE	Wurde der folgende Text von ChatGPT generiert?
ES	¿El siguiente texto fue generado por ChatGPT?

Table 4: Prompts used for AI Feedback.

4.8 Text Vector Features

Our *TextVector* features analyze semantic content of a text, identifying patterns and repetition (Mindner et al., 2023; Shijaku and Canhasi, 2023; Solaiman et al., 2019; Reimers and Gurevych, 2019).

For the features based on Sentence-BERT, we use the sentence-transformer model `distiluse-base-multilingual-cased-v2`[24], since it supports all the languages used in this research. In addition to the four languages in our experiments, it can be used for more than 50 languages, guaranteeing reliable results for possible future research.

4.9 Summary of Our Analyzed Features

Our 8 feature categories contain 37 features. While the *AI feedback* category consists of one feature, the perplexity, semantic, error-based, and readability features each contain two features. The largest feature category are document features, which contains 19 different features. Table 3 summarizes all the features that are part of our experiments.

5 Experimental Setup

In this section, we will describe our experiments with the different feature categories and three classification approaches: The two more traditional approaches XGBoost (Shijaku and Canhasi, 2023) and random forest (RF) (Breiman, 2001) as well as a neural network-based approach with multi-layer perceptrons (MLP) (Murtagh, 1991). As in other studies like Guo et al. (2023); Kumarage et al. (2023); Mitrović et al. (2023), we evaluated the classification performance with accuracy (*Acc*) and F1-score (*F1*). First, we built *text generation detection systems* which were trained, fine-tuned, and tested with our *human-generated* and *AI-generated* texts. Second, we implemented *text rephrasing detection systems* which were trained, fine-tuned, and tested with our *human-generated* and *AI-rephrased* texts. To provide stable results, we used a 5-fold cross-validation, randomly dividing our corpus into 80% training, 10% validation, and 10% unseen test set. The numbers in all tables are the average of the test set results. The best performances are highlighted in bold. As a baseline, we choose two popular state-of-the-art tools which detect texts generated by AI: GPTZero and ZeroGPT. GPTZero is used by over 1 million people (Shrivastava, 2023). However, we found that GPTZero's results were only reliable for *EN* texts. Consequently, we used ZeroGPT as our baseline for *FR*, *DE* and *ES*.

6 Results

Table 5 lists *Acc* and *F1* for detecting *AI-generated* and *-rephrased* texts in *EN*, *FR*, *DE*, and *ES*. For each language classifiers trained on *AI-generated* texts achieve better performances compared to classifiers trained on *AI-rephrased* texts.

6.1 Results of Single Feature Categories

As shown in Figure 1 using the example of $sentiment_{subjectivity}$, the distribution of feature values can differ depending on whether the text is *human-generated*, *AI-generated* or *AI-rephrased* and depending on the language. $sentiment_{subjectivity}$ denotes objectivity (low values) or subjectivity (high values) of a text. Average $sentiment_{subjectivity}$ values tend to be higher for *AI-generated* text than for *human-generated* and *AI-rephrased* text. In general, *DE* texts are the most objective texts—be it *human-* or *AI-generated*—while *EN* and *ES* are more subjective. Moreover, *AI-generated* texts tend to be more subjective than *AI-rephrased* texts for our languages.

[23] https://github.com/textstat/textstat

[24] https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v2

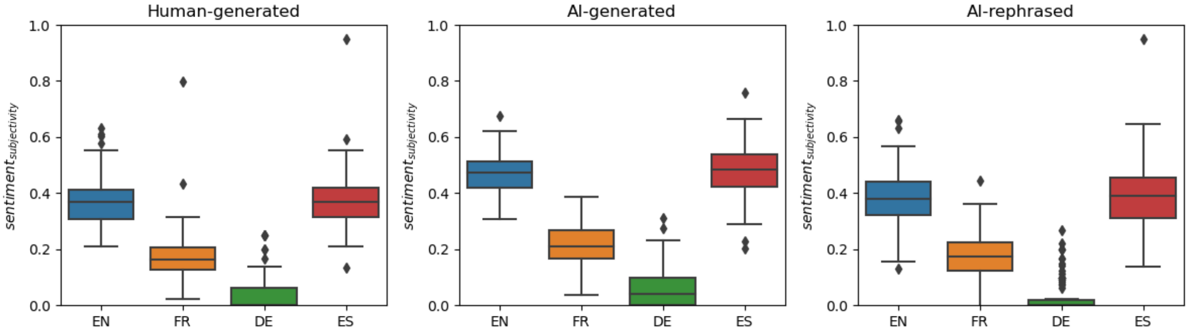


Figure 1: Distribution of $sentiment_{subjectivity}$

		Generated						Rephrased					
		XGBoost		RF		MLP		XGBoost		RF		MLP	
Category	**Lang**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**
Perplexity	EN	83.0	82.2	87.0	85.3	82.0	82.1	52.0	48.7	55.0	54.6	56.0	63.2
	FR	62.0	60.3	69.0	66.8	68.0	69.0	50.0	50.2	53.0	44.2	56.0	58.8
	DE	74.0	74.0	76.0	76.1	81.0	80.6	53.0	53.6	61.0	60.4	56.0	62.7
	ES	82.0	82.3	83.0	82.4	82.0	83.6	56.0	55.4	63.0	63.7	62.0	67.3
Semantic	EN	72.0	72.9	75.0	75.6	73.0	72.3	66.0	64.4	66.0	64.3	52.0	54.3
	FR	61.0	55.8	67.0	65.6	63.0	59.4	55.0	48.2	57.0	50.0	51.0	52.9
	DE	64.0	58.3	64.0	59.8	63.0	63.3	56.0	59.9	54.0	54.4	62.0	60.1
	ES	72.0	69.9	75.0	73.8	76.0	75.7	58.0	56.1	58.0	52.4	53.0	56.3
ListLookup	EN	72.0	72.1	79.0	78.5	71.0	67.8	72.0	73.9	67.0	67.5	69.0	70.3
	FR	72.0	73.0	76.0	76.7	67.0	62.9	66.0	62.6	65.0	65.5	64.0	63.2
	DE	74.0	75.8	79.0	77.8	72.0	74.1	57.0	59.1	58.0	59.2	50.0	52.0
	ES	78.0	79.6	82.0	84.1	73.0	76.8	75.0	75.2	80.0	81.3	77.0	78.4
Document	EN	91.0	91.6	92.0	92.6	87.0	86.0	70.0	69.6	71.0	70.8	78.0	76.1
	FR	94.0	94.2	91.0	90.8	92.0	92.2	86.0	85.3	84.0	80.8	81.0	81.2
	DE	87.0	87.2	90.0	89.6	88.0	88.0	**72.0**	**71.9**	67.0	66.7	71.0	71.3
	ES	96.0	96.2	98.0	98.1	87.0	88.5	84.0	83.4	83.0	82.0	**86.0**	**86.4**
ErrorBased	EN	55.0	61.7	55.0	61.7	56.0	63.9	62.0	68.0	62.0	68.0	62.0	68.0
	FR	62.0	64.2	63.0	67.2	61.0	65.5	53.0	56.0	56.0	58.9	56.0	59.7
	DE	67.0	67.1	67.0	67.1	67.0	69.8	62.0	61.9	62.0	63.5	56.0	50.7
	ES	70.0	71.2	71.0	71.9	71.0	74.6	59.0	56.8	61.0	56.3	64.0	65.2
Readability	EN	60.0	56.3	63.0	59.3	60.0	56.8	54.0	51.1	54.0	47.8	50.0	50.2
	FR	61.0	64.7	62.0	66.0	65.0	67.4	59.0	58.3	60.0	60.6	52.0	31.6
	DE	57.0	53.5	53.0	51.5	57.0	53.6	48.0	41.9	45.0	39.1	45.0	44.9
	ES	74.0	73.7	74.0	72.1	69.0	66.6	54.0	49.1	61.0	50.7	56.0	52.5
AIFeedback	EN	62.0	67.1	62.0	67.1	62.0	68.1	52.0	50.9	50.0	39.8	45.0	30.1
	FR	52.0	24.2	52.0	24.2	48.0	37.2	42.0	33.6	42.0	33.6	55.0	53.4
	DE	49.0	46.1	47.0	35.0	50.0	43.4	52.0	61.8	52.0	61.8	50.0	54.3
	ES	52.0	7.3	52.0	7.3	52.0	20.6	50.0	0.0	52.0	7.3	49.0	25.7
TextVector	EN	90.0	89.9	95.0	94.9	83.0	81.7	**79.0**	**78.2**	75.0	71.0	69.0	65.1
	FR	94.0	94.1	93.0	93.0	85.0	85.4	77.0	77.3	75.0	75.2	68.0	64.2
	DE	87.0	87.0	94.0	94.0	90.0	90.8	68.0	67.5	72.0	67.3	72.0	71.7
	ES	84.0	84.5	91.0	89.5	81.0	76.6	76.0	74.0	76.0	73.6	68.0	64.4
All	EN	90.0	90.9	**98.0**	**98.0**	87.0	87.8	77.0	77.6	71.0	69.8	72.0	71.9
	FR	94.0	94.4	**95.0**	**95.0**	88.0	89.2	**89.0**	**87.9**	86.0	84.2	74.0	66.4
	DE	94.0	93.8	**97.0**	**96.9**	87.0	86.6	70.0	71.6	71.0	68.3	70.0	71.6
	ES	94.0	94.4	**99.0**	**99.0**	90.0	90.2	83.0	82.2	83.0	82.9	78.0	76.1

Table 5: Results for the Detection of *EN FR*, *DE* and *ES AI-generated* and AI-Rephrased Texts.

6.1.1 English

Text Generation Detection The results for *EN* in Table 5 indicate that the system that combines all features (*All*) in an RF performs best (*Acc*=98.0%, *F1*=98.0%). The 2nd-best system is the MLP system that uses *Document* features (*Acc*=95.0%, *F1*=94.9%). The RF system that uses *TextVector* features results in a similar performance (*Acc*=95.0%, *F1*=94.9%). The worst-performing system is the XGBoost system that uses the *ErrorBased* features (*Acc*=55.0%, *F1*=61.7%). Compared to GPTZero ($Acc_{GPTZero}$=76.0%, $F1_{GPTZero}$=78.9%), most of our systems perform better. Our best system with all features (*All*) outperforms GPTZero by 28.9% relative in *Acc* and 24.2% relative in *F1*. ZeroGPT reaches 78.0% $Acc_{ZeroGPT}$ and 81.8% $F1_{ZeroGPT}$. Thus, our best system performs 25.6% relatively better in *Acc*, and 19.8% relatively better in *F1*.

Text Rephrasing Detection The performances for the *EN text rephrasing detection systems* are worse than the *text generation detection systems* for all feature categories except *ErrorBased* (*Acc*=62.0%, *F1*=68.0%). The best-performing system is the XGBoost system that uses *TextVector* features (*Acc*=79.0%, *F1*=78.2%), followed by the MLP system that uses *Document* features (*Acc*=78.0%, *F1*=76.1%). The worst-performing system is the MLP system that uses the *AIFeedback* feature. All our *text rephrasing detection systems* were able to outperform GPTZero ($Acc_{GPTZero}$=43.0% and $F1_{GPTZero}$=27.8%). Our the best-performing *TextVector* feature system even outperforms GPTZero by 83.7% relative in *Acc* and even 159.8% relative in *F1*. ZeroGPT reaches 49.0% $Acc_{ZeroGPT}$ and 43.9% $F1_{ZeroGPT}$. Thus, *Document* outperforms it by 61.2% relative in *Acc* and 81.5% relative in *F1*.

6.1.2 French

Text Generation Detection The results for *FR* in Table 5 demonstrate that the system that combines all features (*All*) in an RF performs best (*Acc*=95.0%, *F1*=95.0%). The 2nd-best system is the XGBoost system that uses *Document* features (*Acc*=86.0%, *F1*=85.3%), followed by the XGBoost system that uses *TextVector* features (*Acc*=77.0%, *F1*=77.3%). The worst-performing systems are those that use the *AIFeedback* feature. Our best *FR* system with all features (*All*) outperforms ZeroGPT ($Acc_{ZeroGPT}$=62.0, $F1_{ZeroGPT}$)=72.6%) by 53.2% relative in *Acc* and 30.9% relative in *F1*.

Text Rephrasing Detection The performances for the *FR text rephrasing detection systems* are worse than the *text generation detection systems* for all feature categories except the MLP system that uses the *AIFeedback* feature (*Acc*=55.0%, *F1*=53.4%). The best-performing system is the system that that combines all features (*All*) in an XGBoost (*Acc*=89.0%, *F1*=87.9%), followed by the XGBoost system that uses *Document* features (*Acc*=86.0%, *F1*=85.3%) and the XGBoost system that uses *TextVector* features (*Acc*=77.0%, *F1*=77.3%). The worst-performing systems are again those that use the *AIFeedback* feature. Our best *FR* system with all features (*All*) outperforms ZeroGPT ($Acc_{ZeroGPT}$=57.0, $F1_{ZeroGPT}$)=67.4%) by 56.1% relative in *Acc* and 30.4% relative in *F1*.

6.1.3 German

Text Generation Detection The results for *DE* in Table 5 indicate that the system that combines all features (*All*) in an RF performs best (*Acc*=97.0%, *F1*=96.9%). The 2nd-best system is the RF system that uses *TextVector* features (*Acc*=94.0%, *F1*=94.0%), followed by the RF system that uses *Document* features (*Acc*=90.0%, *F1*=89.6%). As for the previous languages, the worst-performing systems are those that use the *AIFeedback* feature. Our best *FR* system with all features (*All*) outperforms ZeroGPT ($Acc_{ZeroGPT}$=65.0, $F1_{ZeroGPT}$)=70.9%) by 49.2% relative in *Acc* and 36.7% relative in *F1*.

Text Rephrasing Detection The performances for the *DE text rephrasing detection systems* are worse than the *text generation detection systems* for all feature categories except the systems that use the *AIFeedback* features. The best-performing system is the XGBoost system that that uses the *Document* features (*Acc*=72.0%, *F1*=71.9%), followed by the MLP system that uses *TextVector* features (*Acc*=72.0%, *F1*=71.7%). The worst-performing systems are those that use the *Readability* feature. Our best *DE* system with the *Document* features outperforms ZeroGPT ($Acc_{ZeroGPT}$=48.0, $F1_{ZeroGPT}$=49.5%) by 45.5% relative in *Acc* and 45.3% relative in *F1*.

6.1.4 Spanish

Text Generation Detection The results for *ES* in Table 5 show that the system that combines all features (*All*) in an RF performs best (*Acc*=99.0%, *F1*=99.0%). The 2nd-best system is the RF system that uses *Document* features (*Acc*=98.0%, *F1*=89.1%), followed by the RF system that uses *TextVector* features (*Acc*=91.0%, *F1*=89.5%) and the RF system that uses *ListLookup* features (*Acc*=82.0%, *F1*=84.1%). As for the previous languages, the worst-performing systems are those that use the *AIFeedback* feature. The *F1* of 7.3% is so poor since the feature classifies the text as

AI-generated text in almost all cases. Our best *ES* system with all features (*All*) outperforms ZeroGPT ($Acc_{ZeroGPT}$=60.0, $F1_{ZeroGPT}$)=71.5%) by 65.0% relative in *Acc* and 38.5% relative in *F1*.

Text Rephrasing Detection The performances for the *ES text rephrasing detection systems* are worse than the *text generation detection systems* for all feature categories. The best-performing system is the RF system that uses the *Document* features (*Acc*=86.0%, *F1*=86.4%). The 2nd best system is the system that combines all features (*All*) in an RF (*Acc*=83.0%, *F1*=82.9%), followed by the RF system that uses the *ListLookup* features (*Acc*=80.0%, *F1*=81.3%). The worst-performing systems are those that use the *AIFeedback* feature. The *F1* of 0% and 7.3% are so poor since the feature classifies the text as AI generated text in almost all cases. Our best *ES* system with the *Document* features outperforms ZeroGPT ($Acc_{ZeroGPT}$=52.0, $F1_{ZeroGPT}$=63.7%) by 65.4% relative in *Acc* and 25.6% relative in *F1*.

6.1.5 Combination of All Features

As shown in Table 5, the best performances for the text generation detection systems are achieved using a combination of all features (*All*). Looking at the systems which use all features, the *Acc* for the *AI-generated FR* and *DE* texts is similar with 97.0%, while the *Acc* for the *AI-generated EN* texts is 98.0%. The best *F1* for the *AI-generated DE* classifier is 96.9%. Thus, it is slightly worse than the classifiers trained on our *EN* and *FR* texts which achieved 98.0% and 97.1%, respectively. The best classifier trained on the *AI-generated ES* texts achieved slightly better performances, with 99.0% *Acc* and 99.0% *F1*. Comparing the performances of the systems trained on the *AI-generated* texts, it can be summarized that the classifiers deliver comparable performances across the languages.

The performances of the systems which use all features (*All*) vary more for the *AI-rephrased* texts across the languages. While the best *EN* classifier reaches 79.0% *Acc* on the *AI-rephrased* texts, the best *FR* classifier achieves 89.0% *Acc* on the *AI-rephrased* texts. The *AI-rephrased* detection system for *DE* only achieves 72.0% *Acc*. Compared to the best *DE* text rephrasing detection system, the *FR* system is 23.6% relatively better in *Acc*. The *Acc* for the *ES* text rephrasing detection system is 1% worse than the *FR* system. For *F1*, comparable conclusions can be drawn across the languages. Thus, our investigated features do not deliver comparable performances for the detection of *AI-rephrased* texts across the evaluated languages.

7 Conclusion and Future Work

In this paper, we investigated features to classify whether text is written by a human, generated by AI from scratch or rephrased by AI. We conducted a comparative analysis of the classification across the languages of *EN*, *FR*, *DE*, and *ES*, assessing the performance of these features in their respective linguistic contexts. To train and test classifiers which use the features, we extended the Human-AI-Generated Text Corpus (Mindner et al., 2023)—our new text corpus, which covers 10 different topics for each of the four languages. For *AI-generated text*, our classifier performed best when combining all features, meaning that there are no substantial differences for features across languages. Therefore, we conclude, that the same feature set could also be used for other languages from the same language families. The accuracies are close with 99% for *ES*, 98% for *EN*, 97% for *DE* and 95% for *FR*. In contrast to that, for the detection of AI-rephrased text, the systems with all features outperformed systems with other features in many cases. For *DE* (72%) and *ES* (86%) we achieved the best results using only document features while for *EN* the text vector features yielded the best results (79%).

Although our results indicate that the same feature set could be applied to other languages within the same familie, future work could investigate the applicability of these features across further language families. This would help in understanding the robustness of our method across a more diverse set of languages. Moreover, our corpus currently covers 10 different topics for each language. Extending the corpus to include more topics, and possibly considering different domains and genres, may help in generalizing the findings and making the system more robust. Finally, experimenting with different machine learning architectures such as transformer models could potentially lead to further optimizations.

Ethics Statement

The collected corpus is made freely available to the community. It is based on Wikipedia and news texts. The research was conducted transparently, free from bias and in compliance with applicable laws and regulations. The use of AI models and data is intended to foster a deeper understanding of AI-generated content, with the goal of promoting responsible use and technological innovation.

References

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V Le. 2020. Towards a Human-Like Open-Domain Chatbot. *ArXiv Preprint ArXiv:2001.09977*.

David Arteaga, Juan Arenas, Freddy Paz, Manuel Tupia, and Mariuxi Bruzza. 2019. Design of Information System Architecture for the Recommendation of Tourist Sites in the City of Manta, Ecuador through a Chatbot. In *2019 14th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6. IEEE.

David Baidoo-Anu and Leticia Owusu Ansah. 2023. Education in the Era of Generative Artificial Intelligence (AI): Understanding the Potential Benefits of ChatGPT in Promoting Teaching and Learning. *Available at SSRN 4337484*.

Leo Breiman. 2001. Random Forests. *Machine Learning*, 45(1):5–32.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *CoRR*, abs/2005.14165.

Massimiliano Dibitonto, Katarzyna Leszczynska, Federica Tazzi, and Carlo M Medaglia. 2018. Chatbot in a Campus Environment: Design of LiSA, a Virtual Assistant to Help Students in Their University Life. In *Human-Computer Interaction. Interaction Technologies: 20th International Conference, HCI International 2018, Las Vegas, NV, USA, July 15–20, 2018, Proceedings, Part III 20*, pages 103–116. Springer.

Ethnologue. 2023. What are the top 200 most spoken languages?

Clara Falala-Séchet, Lee Antoine, Igor Thiriez, and Catherine Bungener. 2019. OWLIE: A Chatbot that Provides Emotional Support for Coping With Psychological Difficulties. In *Proceedings of the 19th ACM International Conference on Intelligent Virtual Agents*, pages 236–237.

Rudolf Franz Flesch. 1948. A New Readability Yardstick. *The Journal of applied psychology*, 32 3:221–233.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical Detection and Visualization of Generated Text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116, Florence, Italy. Association for Computational Linguistics.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection.

Katharina Jeblick, Balthasar Schachtner, Jakob Dexl, Andreas Mittermeier, Anna Theresa Stüber, Johanna Topalis, Tobias Weber, Philipp Wesp, Bastian Sabel, Jens Ricke, and Michael Ingrisch. 2022. ChatGPT Makes Medicine Easy to Swallow: An Exploratory Case Study on Simplified Radiology Reports. *ArXiv E-Prints*.

Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Xing Wang, and Zhaopeng Tu. 2023. Is ChatGPT a Good Translator? A Preliminary Study. *ArXiv Preprint ArXiv:2301.08745*.

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*, pages 4171–4186.

J. Peter Kincaid, Robert P. Fishburne, Richard L. Rogers, and Brad S. Chissom. 1975. Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel.

Tharindu Kumarage, Joshua Garland, Amrita Bhattacharjee, Kirill Trapeznikov, Scott Ruston, and Huan Liu. 2023. Stylometric Detection of AI-Generated Text in Twitter Timelines.

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.

Bertalan Mesko. 2023. The chatgpt (generative artificial intelligence) revolution has made artificial intelligence approachable for medical professionals. *Journal of medical Internet research*, 25:e48392.

Lorenz Mindner, Tim Schlippe, and Kristina Schaaff. 2023. Classification of Human- and AI-Generated Texts: Investigating Features for ChatGPT. *TBD*.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature. In *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*.

Sandra Mitrović, Davide Andreoletti, and Omran Ayoub. 2023. ChatGPT or Human? Detect and Explain. Explaining Decisions of Machine Learning Model for Detecting Short ChatGPT-Generated Text. *arXiv preprint arXiv:2301.13852*.

Fionn Murtagh. 1991. Multilayer Perceptrons for Classification and Regression. *Neurocomputing*, 2(5):183–197.

Corina Pelau, Dan-Cristian Dabija, and Irina Ene. 2021. What Makes an AI Device Human-Like? The Role of Interaction Quality, Empathy and Perceived Psychological Anthropomorphic Characteristics in the Acceptance of Artificial Intelligence in the Service Industry. *Computers in Human Behavior*, 122:106855.

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

Adam Roberts, Colin Raffel, Katherine Lee, Michael Matena, Noam Shazeer, Peter J. Liu, Sharan Narang, Wei Li, and Yanqi Zhou. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. Technical report, Google.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter. In *Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition (EMC2-NIPS 2019)*.

Rexhep Shijaku and Ercan Canhasi. 2023. ChatGPT Generated Text Detection.

Rashi Shrivastava. 2023. With Seed Funding Secured, AI Detection Tool GPTZero Launches New Browser Plugin.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. 2019. Release Strategies and the Social Impacts of Language Models.

Mayank Soni and Vincent Wade. 2023. Comparing Abstractive Summaries Generated by ChatGPT to Real Summaries Through Blinded Reviewers and Text Classification Algorithms.

Viriya Taecharungroj. 2023. “What Can ChatGPT Do?” Analyzing Early Reactions to the Innovative AI Chatbot on Twitter. *Big Data and Cognitive Computing*, 7(1):35.

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.

Peipeng Yu, Jiahan Chen, Xuan Feng, and Zhihua Xia. 2023. CHEAT: A Large-scale Dataset for Detecting ChatGPT-writtEn AbsTracts.

Wataru Zaitsu and Mingzhe Jin. 2023. Distinguishing ChatGPT(-3.5, -4)-generated and Human-Written Papers Through Japanese Stylometric Analysis.

Handling Realistic Label Noise in BERT Text Classification

Maha Tufail Agro
MBZUAI
maha.agro@mbzuai.ac.ae

Hanan Aldarmaki
MBZUAI
hanan.aldarmaki@mbzuai.ac.ae

Abstract

Label noise refers to errors in training labels caused by cheap data annotation methods, such as web scraping or crowd-sourcing, which can be detrimental to the performance of supervised classifiers. Several methods have been proposed to counteract the effect of random label noise in supervised classification, and some studies have shown that BERT is already robust against high rates of randomly injected label noise. However, real label noise is not random; rather, it is often correlated with input features or other annotator-specific factors. In this paper, we evaluate BERT in the presence of two types of realistic label noise: feature-dependent label noise, and synthetic label noise from annotator disagreements. We show that the presence of these types of noise significantly degrades BERT classification performance. To improve robustness, we evaluate different types of ensembles and noise-cleaning methods and compare their effectiveness against label noise across different datasets.

1 Introduction

Deep learning algorithms have been shown to perform extremely well in supervised classification tasks given high-quality datasets. Unfortunately, obtaining gold-standard labels is often prohibitively expensive with large-scale datasets, leading practitioners to resort to cheaper data collection methods such as crowd-sourcing or automatic annotation methods (Yan et al., 2014). These techniques are known to impart a substantial amount of label noise in the data, which can degrade classification performance (Ji et al., 2021). Label noise refers to errors or inconsistencies within the data labels, such that the prescribed labels do not match the gold labels assigned by experts. Datasets obtained through web scraping often contain label noise given the absence of expert-verified gold labels (Li et al., 2017). Due to a meteoric rise in social media usage, more and more datasets are automatically acquired from online social platforms, and such datasets are likely to contain label noise. Small-scale datasets can also suffer from the same problem if the annotation process is challenging or the annotators have divergent opinions (Ma et al., 2019).

Some prior works have been dedicated to developing and deploying algorithms that combat the effects of label noise in text classification (Han et al., 2018; Sukhbaatar et al.; Zhang and Sabuncu, 2018; Jiang et al., 2018). However, most previous studies simulated label noise by random substitution, and recent research has shown empirically that many methods that successfully handle random noise are ineffective against real-world label noise (Jiang et al., 2020). In the text classification domain, Zhu et al. (2022) explored the robustness of previously proposed methods for handling label noise, including noise matrix with regularization (Jindal et al., 2019), co-teaching (Han et al., 2018), and label smoothing (Szegedy et al., 2016). They concluded that BERT (Devlin et al., 2019) is already robust against randomly injected label noise and these approaches obtain no additional performance gains. On the other hand, they find that feature-dependent label noise, which realistically arises from automatic annotation techniques, degrades BERT performance and these noise handling techniques add little to no robustness at all. This creates a need for a comprehensive evaluation of noise-robust methods in the domain of text classification, considering the presence of realistic labeling errors.

In this paper, we describe methods and experiments for handling realistic label noise in BERT text classification. We use two datasets that contain feature-dependent label noise from automatic annotation, namely Yorùbá and Hausa (Hedderich et al.). These two datasets have been manually cleaned, so a clean version of each exists for evaluation. In addition, we use tweetNLP (Gimpel et al., 2011) and SNLI (Bowman et al., 2015) datasets with synthetic noise that mimics human errors by utilizing multiple crowd-sourced annotations (Chong et al., 2022). This collection of datasets provides a range of noise types and levels that more closely reflect realistic label noise compared to random noise injection. We evaluate the performance of vanilla BERT compared with a subset of noise-handling approaches, namely co-teaching (Han et al., 2018), Consensus Enhanced Training Approach (CETA) (Liu et al., 2022), different types of ensembles (Ganaie et al., 2022), and noise cleaning (Chong et al., 2022; Sluban et al., 2014). We summarize our findings as follows:

1. For datasets with feature-dependent label noise, we find that CETA, some types of ensembles, and noise cleaning, all provide positive performance gains compared to vanilla BERT.

2. For synthetic label noise from multiple annotations, we do not observe significant gains using these approaches. We surmise that this type of noise is more challenging or may even reflect inherently ambiguous labels.

It is worth noting that the noise is qualitatively different in these two categories of label noise as the latter arises from human rather than automatic processes, which could be due to either errors or genuine disagreements. Some recent works attempt to include multiple labels in the training process rather than rely on a single gold label to account for the inherent uncertainty from human disagreements. This may be justified given the nature of some tasks, and the noising scheme performed on tweetNLP and SNLI may warrant that kind of treatment or further scrutiny to identify clear-cut errors. However, as we focus on noise robustness as the scope of this work, we treat the synthetic noise int these datasets as labeling errors and leave any further analysis of this sort for future work.

2 Background & Related Works

2.1 Types of Label Noise

Label noise refers to irregularities or inconsistencies within the data labels, where the prescribed label of a data point does not correspond to the true expert label. In other words, noisy instances in this context specifically pertain to inaccuracies or errors in the labeling of the data, rather than any distortions or imperfections in the input data itself.

When observing the effect of label noise, the majority of existing literature in text classification assumes random injection of label noise (Han et al., 2018; Sukhbaatar et al.; Zhang and Sabuncu, 2018). This type of synthetic noising involves randomly permuting a fixed number of labels according to a pre-defined noise level and noise type. Because the process of simulating such noise is entirely random and does not depend on the input data features in any way, this type of noise is also known as *feature-independent* label noise.

In contrast, *feature-dependent* label noise is correlated with input features (Algan and Ulusoy, 2020). Datasets that use distantly or weakly supervised methods to generate labels are prone to this type of label noise. These approaches are often used in low-resource applications where it is impractical or expensive to manually annotate large amounts of data. Relation extraction is one such application that heavily relies on automatic data generation methods as supervised relation extraction methods necessitate an extensive amount of labeled training data (Mintz et al., 2009). In this area, denoising methods such as the ones proposed in Jia et al. (2019), Qin et al. (2018), Liu et al. (2022) and Ma et al. (2021) are specifically developed to address feature-dependent label noise in relation extraction datasets.

Recently, Chong et al. (2022) developed realistic noising methods that mimic how humans make labeling errors by taking advantage of the multiple rounds of annotation that some datasets undergo. During the annotation process, certain subsets of the data are subjected to rigorous validation schemes, such as gold labels assigned by experts, while others are annotated using less stringent methods, such as crowdsourced evaluations. By incorporating varying annotations generated during this process, their approach produces realistic label noise that reflects how humans make errors. We refer to this noising scheme as *pseudo-real-world label* noise.

2.2 Noise-robust methods

Noise-robust methods in the literature include model enhancements such as **robust loss functions**. Robust loss functions are a class of loss functions used to train models in a way that is more resistant to label noise. One such loss function is the family of generalized cross-entropy loss functions (Zhang and Sabuncu, 2018), which are designed to be more robust to label noise by penalizing the model less for incorrect predictions that are consistent with noisy labels.

Another class of noise-robust approaches is what we refer to as **multi-netowrk training**. This subcategory of methods introduces multiple networks that learn from each other and as such make more informed decisions regarding which data to use to update the model parameters. For instance, co-teaching (Han et al., 2018) includes two models that are trained in parallel, and each model is presented with examples that incur low loss by the other model. Intuitively, correct labels produce small losses in earlier training epochs and noisy labels produce higher losses. Similarly, the Consensus-Enhanced Training Approach (CETA) proposed in Liu et al. (2022) is a methodology for robust sentence-level relation extraction that emphasizes the selection of clean data points during the training process. The denoising technique is applied to establish a robust boundary for classification, preventing inaccurately labeled data from being assigned to the wrong classification space, and the consensus between two divergent classifiers is used to select clean instances for training.

2.3 Noise cleaning approaches

Noise-cleaning aims to automatically segregate clean data from noisy data in order to train the final classifier using a cleaned subset of the original training set. The “small loss trick" is commonly used to identify potentially noisy or mislabeled data. The intuition behind this approach is that noisy data have comparatively higher loss than clean data (Takeda et al., 2021; Han et al., 2018; Jiang et al., 2018; Ji et al., 2021).

Several approaches have been proposed for automatic noise detection, which can be a first step towards noise-cleaning before training a robust classifier. Wheway (2001) used boosting to detect noisy data instances. The approach involves iteratively re-weighing the data points to emphasize those that are most difficult to classify correctly. The resulting model is then used to identify the noisy data points by measuring their contribution to the final model. Sluban et al. (2014) trained multiple classifiers (ensemble) on different subsets of the data and combined their outputs to obtain a noise ranking for each instance. Similarly, Chong et al. (2022) assessed the performance of pre-trained language models as error detectors using clean held-out data. They experiment with the error detection capabilities of individual pre-trained models and an ensemble of pre-trained language models. They find that an ensemble of pre-trained model losses outperforms individual model loss in error detection.

2.4 Label noise & BERT

BERT (Devlin et al., 2019) is a popular pre-trained language model that is frequently used for text classification by fine-tuning on target labels. Some recent studies have shown that BERT is already robust against randomly injected label noise (Zhu et al., 2022), and early stopping is sufficient to prevent overfitting on noisy labels. Zhu et al. (2022) evaluates popular noise robust approaches in BERT text classification such as appending noise transition matrix after BERT’s predictions (Sukhbaatar et al.), acquiring the noise transition matrix with $l2$ regularization (Jindal et al., 2019), and multi-network training via co-teaching (Han et al., 2018). They conclude that while BERT appears to be inherently robust to feature-independent noise, none of the above approaches improves BERT’s peak performance in the presence of feature-dependent label nose.

3 Methodology

In this work, we evaluate BERT text classification on datasets containing pseudo-real-world label noise and feature-dependent label noise. We do not consider randomly injected label noise as Zhu et al. (2022) have shown BERT to be already robust to this type of synthetic noise. The scope of this work is limited to text classification with BERT following the baselines established by Zhu et al. (2022).

3.1 Datasets

To study feature-dependent label noise, we use two news-topic categorization datasets from two low-resource African languages: Hausa and Yorùbá (Hedderich et al.). These languages are spoken by large populations in Africa, with Hausa being the second most spoken indigenous language, with 40 million native speakers, and Yorùbá being the third most spoken, with 35 million native speakers[1]. For these datasets, gazetteers were used for automatic labeling, which results in feature-dependent label noise. For instance, when identifying texts for the “Africa" class, a labeling rule based on a list of

[1] https://en.wikipedia.org/wiki/Languages_of_Africa

Dataset	Yorùbá	Hausa	TweetNLP	SNLI
Number of classes	7	5	15	3
Average sentence length	13	10	12	21
Train Samples	1340	2045	11565	363043
Validation Samples	189	290	2874	9831
Test Samples	379	582	-	9815
Train Noise Level	33.28%	50.37%	Various	Various

Table 1: Dataset statistics

African countries and their capitals was employed. These datasets were chosen specifically as they contain automatic annotation label noise i.e., weak-supervision/feature-dependent noise in addition to clean versions of the splits, making it possible to establish ground truth. Note that the amount of label noise in Hausa and Yorùbá is fixed.

Furthermore, we use the noising schemes proposed by Chong et al. (2022) to simulate real-world label noise produced by crowd-sourced labeling. Pseudo-real-world label noise is injected in two benchmark datasets: TweetNLP (Gimpel et al., 2011) and Stanford Natural Language Inference (SNLI) dataset (Bowman et al., 2015). TweetNLP is a part-of-speech tagging dataset developed by scraping Twitter posts. While TweetNLP already contained crowd-sourced labels, it later received separate crowdsource evaluations, allowing access to multiple annotations from separate annotators. The SNLI dataset is a large Natural Language Inference corpus developed at Stanford. The original corpus consists of 570K sentence pairs, manually labeled by experts. Like TweetNLP, a subset of SNLI later received extensive crowdsource evaluation. We noise both TweetNLP and SNLI to three label noise levels: 10%, 20%, and 30%. Data statistics for all datasets are shown in Table 1.

3.2 Baselines

Zhu et al. (2022) already evaluated the noise matrix approach (Sukhbaatar et al.), label smoothing (Szegedy et al., 2016), and co-teaching (Han et al., 2018) on the feature-dependent datasets, Hausa and Yorùbá, and concluded that no gains are observed using these methods. We use the following as baselines to benchmark our experiments using other approaches:

1. **Vanilla BERT**: BERT trained on noisy training data without a noise-handling mechanism, except early stopping on a noisy validation set, as done in Zhu et al. (2022).
2. **Co-teaching** (Han et al., 2018), which simultaneously trains two networks, with each network independently ranking data points based on their loss to guide the other network on which points to be included for training. In other words, each network independently performs noise-cleaning for the other network.

4 Approaches

We experiment with the following approaches as potential methods for improving performance under realistic label noise conditions:

4.1 Consensus-Enhanced Training Approach (CETA)

CETA (Liu et al., 2022) has been proposed as a noise-robust model for relation extraction and has shown promising results. CETA contains two discrepant classifiers that share a single encoder. The focus of CETA is to train the classifiers only in instances where both classifiers have reached a consensus. Such instances are supposedly deemed clean. To achieve consensus, CETA augments the standard cross entropy loss to include predictions from both classifiers and uses the Wasserstein distance (Kantorovich, 2006) as a secondary criterion. In this manner, CETA can also be considered an ensemble learning approach.

4.2 Deep Ensembles

Deep ensembles have been shown to generally exhibit robustness as compared to singular models and reduce overfitting (Ganaie et al., 2022). To that end, we hypothesize that ensembles may excel in noisy classification tasks due to the presence of label noise in the training data, which can cause individual models to learn false correlations between features and labels. By training multiple classifiers and combining their predictions, each model can develop a unique representation of the input data and filter out spurious information, leading to a more robust classification boundary. While ensembles have been previously proposed for data and label noise detection (Wheway, 2001; Sluban et al., 2014; Chong et al., 2022), their performance as a method of robust text classification with noisy labels has not been established.

We formally define ensembles as follows: Given m classifiers $C_1, C_2, ..., C_m$, each classifier produces probabilities P_{c_i} on a clean test set T, an ensemble of the predictors averages the probabilities of each predictor such that $P_{ensemble} = \sum_{i=1}^{m} \frac{P_{C_i}}{m}$. It should be noted that each ensemble member is trained on either the same noisy training set or a randomly selected subset of the noisy training set, depending on the employed technique, which is described below. Nevertheless, in all scenarios, each member is evaluated on the same clean test set. We experiment with three types of ensembles:

1. **Homogeneous Ensembles** Ensembles that aggregate predictions from the same type of classifier (i.e. vanilla BERT with early stopping), trained with different initializations and hyperparameters.

2. **Heterogeneous Ensembles** Ensembles that aggregate predictions from different types of classifiers. In our experiments, we use vanilla BERT, co-teaching, and CETA as the heterogeneous classifiers in the ensemble.

3. **Boosting** Ensembles that aggregate predictions from the same type of classifier (i.e. vanilla BERT with early stopping), but each classifier is trained on a different subset of the training data.

4.3 Noise Cleaning Based on Fine-Tuned Model Loss

We use the pre-trained language model's ability to identify noisy labels as a way to clean the training set by removing instances with potential label noise. This involves fine-tuning BERT on noisy task-specific training data and evaluating model loss on each instance. Training instances that have a loss higher than the selected threshold are excluded from the training set used to train the final classifier. We tune the loss threshold on a noisy validation set.

To avoid biasing or overfitting the model when computing loss on the same set used for fine-tuning, we employ an N-fold process to calculate the loss only on held-out data points[2]. The process is outlined in Algorithm 1. In summary, we fine-tune the model using a subset of the noisy training set and use the model to identify and remove noisy samples from the held-out validation set using a fixed loss threshold[3]. The process is repeated separately N times using disjoint validation sets to clean the complete training set.

Algorithm 1 Noise Cleaning Algorithm

```
Input: Noisy train set $T$, loss threshold $t$, number of folds $f$
Output: Cleaned train set $T_{\text{clean}}$
Divide $T$ into $f$ validation subsets: $V_1, \ldots, V_f$
for $i = 1$ to $f$ do
    $T_i = T \setminus V_i$
    Train a fine-tuned model $M_i$ on $T_i$
    Evaluate the model loss $L_{V_i}$ on $V_i$
    $T_{\text{clean},i} = V_i[L_{V_i} < t]$
end for
$T_{\text{clean}} = \bigcup_{i=1}^{f} T_{clean,i}$
return $T_{\text{clean}}$
```

5 Experiments and Results

All of the methods evaluated in these experiments incorporate early stopping on noisy validation set as done by Zhu et al. (2022). We use a noisy validation set because obtaining a clean validation set is often not feasible in practice. Moreover, Zhu et al. (2022) show that using a noisy validation set for early stopping is more or less as effective as using a clean validation set.

5.1 Hyperparameters

The number of training steps is optimally set to 3000[4] unless we are required to vary hyperparameter settings for homogeneous ensembles. For homogeneous ensembles, we cycle through a combination of the following hyperparameters: the number of training steps = [2000, 3000, 4000, 5000, 6000], learning rate = [0.0002, 0.0004, 0.0005, 0.00001, 0.00002, 0.00003, 0.00004, 0.00005], patience (for early stopping) = [25, 30, 40, 50], warm-up steps =

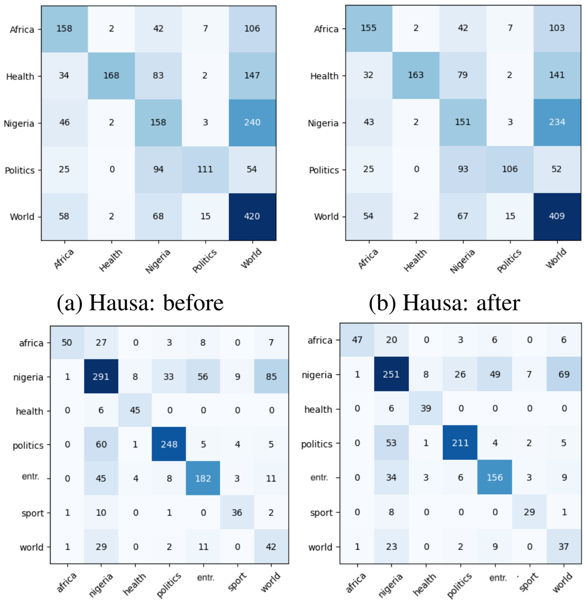
[2] A similar approach is briefly described in (Northcutt et al., 2021) for estimating noise characterization in the confident learning framework.

[3] The loss threshold is a hyperparameter that we tune beforehand.

[4] If the validation accuracy does not improve beyond a certain patience level, we employ early stopping to prematurely halt the training process for all experiments.

	Hausa	Yorùbá
Clean Data		
Vanilla BERT	82.67 ± 0.73	76.23 ± 0.28
Noisy Data		
Vanilla BERT	46.98 ± 1.01	64.72 ± 1.45
Co-Teaching	**48.11** ± 1.71	64.38 ± 0.98
CETA	*__49.31__ ± 0.31	*__68.07__ ± 0.18
HME	46.39 ± 0.21	**67.28** ± 0.81
HTE	**48.28** ± 0.19	**67.81** ± 0.73
Boosting	**47.13** ± 0.42	**67.63** ± 1.26
NC	**47.18** ± 0.22	62.17 ± 0.54

Table 2: A comparison of proposed methods against baselines on datasets with feature-dependent label noise. **HME**: Homogeneous ensembles **HTE**: Heterogeneous ensembles. Boosting: Ensembles of different random subsets from the train set. **NC**: Noise Cleaning. Average accuracy is reported with a standard deviation from 5 runs of each experiment.



(a) Hausa: before

	Africa	Health	Nigeria	Politics	World
Africa	158	2	42	7	106
Health	34	168	83	2	147
Nigeria	46	2	158	3	240
Politics	25	0	94	111	54
World	58	2	68	15	420

(b) Hausa: after

	Africa	Health	Nigeria	Politics	World
Africa	155	2	42	7	103
Health	32	163	79	2	141
Nigeria	43	2	151	3	234
Politics	25	0	93	106	52
World	54	2	67	15	409

(c) Yorùbá: before

	africa	nigeria	health	politics	entr.	sport	world
africa	50	27	0	3	8	0	7
nigeria	1	291	8	33	56	9	85
health	0	6	45	0	0	0	0
politics	0	60	1	248	5	4	5
entr.	0	45	4	8	182	3	11
sport	1	10	0	1	0	36	2
world	1	29	0	2	11	0	42

(d) Yorùbá: after

	africa	nigeria	health	politics	entr.	sport	world
africa	47	20	0	3	6	0	6
nigeria	1	251	8	26	49	7	69
health	0	6	39	0	0	0	0
politics	0	53	1	211	4	2	5
entr.	0	34	3	6	156	3	9
sport	0	8	0	0	0	29	1
world	1	23	0	2	9	0	37

Figure 1: Noise matrices for Hausa and Yorùbá showing noise distribution before and after noise cleaning.

[0, 1, 5, 7, 10], weight decay = [0.1, 0.001, 0.0001], and drop rate = [0.1, 0.25, 0.5, 0.8].

For other experiments that do not explicitly require us to vary hyperparameters, we fix the following hyperparameters for the African language datasets, training steps = 3000, learning rate = 0.00005, patience = 25, drop rate = 0.1, warm-up steps = 0, weight decay = 0.1. We fix the following hyperparameters for the English language datasets, training steps = 3000, learning rate = 0.00002, patience = 25, drop rate = 0.25, warm-up steps = 0, weight decay = 0.1. For boosting related experiments, we experiment with two training data subset sizes: 50% of the total training data and 80% of the total training data. For heterogeneous ensembles, we aggregate predictions from the following three classifiers: vanilla BERT, co-teaching, and CETA.

5.2 BERT Models

We use *bert-base-uncased*[5] as the backbone for our English language datasets: TweetNLP and SNLI. We use *bert-base-multilingual-cased*[6] for our African language datasets: Yorùbá and Hausa.

5.3 Loss threshold

To select a loss threshold for noise-cleaning as described in section 4.3, we experiment with different cut-off points in the following interval $[6.0, 8.0]$. We use only a noisy validation set to select the loss threshold. Data points whose loss exceeds the fixed loss threshold are excluded from the training set, effectively 'cleaning' the noisy training set to a certain extent. Note that we only report results on the loss threshold that produces the most optimal accuracy on the noisy validation set. The cleaned training set is once again used to train a vanilla BERT model, at which point we can evaluate how well the noising scheme performed.

5.4 Results

5.5 Feature-dependent label noise

Table 2 shows the results of baseline models and the proposed approaches on datasets containing feature-dependent label noise: Hausa and Yorùbá.

First, we observe that co-teaching and noise cleaning do not consistently improve performance compared to vanilla BERT. CETA, on the other hand, improves performance by around 3 absolute percentage points on both datasets. The homogeneous ensemble method does not consistently improve either, but we do observe consistent gains using heterogeneous ensembles and boosting.

Figure 1 show the noise distribution in the training set before and after applying the noise cleaning procedure in both datasets. Note that the noise-cleaning method results in the removal of both noisy and clean instances, which leads to the total noise level not being considerably reduced. Overall, we we do not observe a larger reduction in noise level in either dataset. After noise cleaning, we

[5] https://huggingface.co/bert-base-uncased

[6] https://huggingface.co/bert-base-multilingual-cased

	TweetNLP			SNLI		
Noise Level	10%	20%	30%	10%	20%	30%
Clean Data						
Vanilla BERT	91.03 ± 0.81			85.03 ± 0.16		
Noisy Data						
Vanilla BERT	82.08 ± 0.03	74.45 ± 0.65	72.96 ± 1.42	84.79 ± 0.87	**83.83** ± 1.01	82.01 ± 0.21
Co-Teaching	81.31 ± 0.11	73.68 ± 0.04	72.41 ± 0.71	84.27 ± 0.15	83.10 ± 1.20	80.99 ± 0.04
CETA	81.00 ± 1.81	72.40 ± 1.01	72.13 ± 0.71	84.24 ± 0.01	82.67 ± 0.21	81.02 ± 0.27
HME	81.81 ± 0.05	74.08 ± 0.03	72.53 ± 0.01	85.02 ± 0.12	83.76 ± 0.10	81.99 ± 0.26
HTE	79.13 ± 0.32	**74.90** ± 0.51	72.32 ± 0.97	84.75 ± 0.34	83.64 ± 1.11	81.16 ± 0.97
Boosting	**82.53** ± 0.01	74.27 ± 0.15	**73.52** ± 3.32	**85.38** ± 0.45	83.80 ± 0.81	**82.06** ± 0.41
NC	80.94 ± 0.09	74.55 ± 0.45	72.65 ± 0.19	85.13 ± 0.05	**84.00** ± 0.01	**82.97** ± 1.09

Table 3: A comparison of proposed methods against baselines on TweetNLP and SNLI datasets noised to various levels. HME: Homogeneous ensembles HTE: Heterogeneous ensembles. Boosting: Ensembles of different random subsets from the training set. Average accuracy is reported with the standard deviation from 5 runs of each experiment.

have 31% label noise in Yorùbá compared to 33% before noise cleaning, with only a 2% reduction in noise. For Hausa, the noise level after cleaning is similarly reduced by 3% (47% compared to 50% before cleaning). In summary, we do not find the noise-cleaning method to be an efficient error detector for feature-dependent label noise, as compared to the other noise-robust we use. This is inconsistent with the result in Chong et al. (2022), where they show that language models are suitable for label error detection. However, they also report that an ***ensemble of large*** pre-trained language models is a better error detector than a smaller individual pre-trained model, and in both cases, while models may have good error detection performance, the performance in the underlying task is not necessarily improved.

5.6 Pseudo-real-world label noise

Table 3 shows the results on datasets containing pseudo-real-world label noise, TweetNLP, and SNLI, with three levels: 10%, 20%, and 30%. In these datasets, we observe that performance drops significantly with increased noise levels in TweetNLP, but only small drops in performance are observed in SNLI. We hypothesize that this potentially reflects the inherent difficulty in the natural language inference task, and the gold labels may already be ambiguous even before applying the noising scheme. Table 4 shows samples from both SNLI and TweetNLP datasets before and after injecting noisy labels. In many cases, particularly in SNLI, the given example is rather ambiguous and both labels can be suitable. These are also cases where there are high inter-annotator disagreements.

In terms of noise handling techniques, we observe that all approaches generally do not produce large gains in performance compared to vanilla BERT. Furthermore, many approaches result in slightly worse performance compared to the baseline. Boosting seems like the most robust technique, as it maintains baseline performance at least, while also being effective against feature-dependent label noise. Noise cleaning in this category obtained mixed results. Surprisingly, CETA does not excel over other methods in this particular category. Although it was specifically designed to address feature-dependent label noise, its performance is somewhat inferior to the vanilla BERT baseline when dealing with realistic label noise. We surmise that this type of artificial noise is more challenging as it's based on actual human errors, and may even reflect intrinsic ambiguities in the task, which makes it harder to detect through automatic approaches.

6 Conclusions

In this paper, we described experiments for evaluating different label noise handling techniques within the framework of BERT text classification. We evaluated some multi-network training approaches (i.e. co-teaching and CETA), different types of ensembles (homogeneous, heterogeneous, and boosting), and a noise cleaning technique and compared them with a vanilla BERT fine-tuned model with early stopping. We used two datasets that contain feature-dependent label noise from automatic labeling, as well as two datasets with synthetic pseudo-real-world label noise obtained by considering multiple

Dataset	Text	Noisy Label	Actual Label
SNLI	(1) Young man wearing a blue jacket, green shirt and denim jeans is photographed by person in beige jacket and burgundy pants while four onlookers watch on an expanse of sand.<!SEP!> The people are ignoring the man getting photographed.	No Relationship	~~Contradiction~~
SNLI	(2) A man wearing a black t-shirt is playing seven string bass a stage.<!SEP!> The man is playing an old guitar.	Contradiction	~~No Relationship~~
SNLI	(3) Many children are sitting in a classroom watching a woman in the front.<!SEP!>The woman is teaching the children	Entailment	~~No Relationship~~
TweetNLP	(1) Reading harry potter in bed! waiting for the new south park to come on	ADJ	~~NOUN~~
TweetNLP	(2) @USER: I'm not insulted, at all, trust me. I'm seeking to understand you and your video. :)	DET	~~ADP~~
TweetNLP	(3) Chicagoan early voters in Uptown even get brownies and entertainment while waiting for a dozen people to do number page ballots.	ADJ	~~NOUN~~

Table 4: Samples from SNLI and TweetNLP with pseudo-real-world noise injection, highlighting the complexity and potential ambiguity of these tasks.

annotations.

For feature-dependent label noise, the recently proposed Consensus Enhanced Training Approach (CETA) shows the most promising results compared to the baselines. Some ensembling techniques, such as boosting, can also improve performance compared to the baselines but do not provide the level of robustness achieved via CETA.

While pre-trained language models have been shown previously to have the potential to detect label errors through out-of-sample loss, our results indicate that using this technique to automatically clean the data does not result in improved performance compared to using the noisy set. This may suggest that removing label errors is not necessarily a good approach for handling label noise; rather, error detection can be used to identify noisy labels for manual correction.

The synthetic pseudo-real-world category of label noise appears to be more challenging as the noise represents actual human errors, which could be an indication of inherent ambiguities in the task itself. Our experiments show that most techniques do not improve performance compared to the baselines. Furthermore, for a dataset like SNLI, which is known to be challenging even for human annotators, the presence of label noise does not reduce the performance to a great extent compared to the other datasets. This may suggest that the noising scheme is compatible with the inherent difficulty or label ambiguity of the task, and any attempts to detect or discard the noise will not necessarily improve the performance using stringent metrics such as accuracy. Recent efforts to embrace annotator disagreements and incorporate them in the training process (Zhang et al., 2021) rather than relying on a single gold label may be more suitable to handle this kind of labeling inconsistencies.

Overall, the results indicate that handling realistic label noise in text classification remains a challenging task, and none of the noise-handling techniques examined so far has shown consistent performance improvements across multiple datasets.

Limitations

The work described in this paper is limited by the small number of datasets that contain both noisy and clean versions in the text classification domain, which are needed for evaluating noise-handling methods. While we observed positive results from at least two approaches, any conclusions we make about their effectiveness are drawn from a sample of two datasets, and may not necessarily generalize to other cases. For the pseudo-real-world label noise category, it is unclear whether the noise represents true errors or inherent ambiguity in the task. The mixed results we observe could also be a result of ambiguities in the presumed 'clean' test set.

References

Gorkem Algan and Ilkay Ulusoy. 2020. Label noise types and their effects on deep learning. *ArXiv*, abs/2003.10471.

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 (EMNLP)*, page unknown. Association for Computational Linguistics.

Derek Chong, Jenny Hong, and Christopher Manning. 2022. Detecting label errors by using pre-trained language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9074–9091. 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.

M.A. Ganaie, Minghui Hu, A.K. Malik, M. Tanveer, and P.N. Suganthan. 2022. Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, 115:105151.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*, pages 42–47. The Association for Computer Linguistics.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 8536–8546, Red Hook, NY, USA. Curran Associates Inc.

Michael A. Hedderich, David Adelani, Dawei Zhu, Jesujoba Alabi, Udia Markus, and Dietrich Klakow. Transfer learning and distant supervision for multilingual transformer models: A study on African languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2580–2591. Association for Computational Linguistics.

Daehyun Ji, Dokwan Oh, Yoonsuk Hyun, Oh-Min Kwon, and Myeong-Jin Park. 2021. How to handle noisy labels for robust learning from uncertainty. *Neural Networks*, 143:209–217.

Wei Jia, Dai Dai, Xinyan Xiao, and Hua Wu. 2019. ARNOR: Attention regularization based noise reduction for distant supervision relation classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1399–1408, Florence, Italy. Association for Computational Linguistics.

Lu Jiang, Di Huang, Mason Liu, and Weilong Yang. 2020. Beyond synthetic noise: Deep learning on controlled noisy labels. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4804–4815. PMLR.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2304–2313. PMLR.

Ishan Jindal, Daniel Pressel, Brian Lester, and Matthew Nokleby. 2019. An effective label noise model for DNN text classification. 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 3246–3256, Minneapolis, Minnesota. Association for Computational Linguistics.

Leonid V Kantorovich. 2006. On the translocation of masses. *Journal of mathematical sciences*, 133(4):1381–1382.

Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L. Li. 2017. Learning from noisy labels with distillation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1928–1936. IEEE Computer Society.

Ruri Liu, Shasha Mo, Jianwei Niu, and Shengda Fan. 2022. CETA: A consensus enhanced training approach for denoising in distantly supervised relation extraction. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2247–2258, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Kede Ma, Xuelin Liu, Yuming Fang, and Eero P. Simoncelli. 2019. Blind image quality assessment by learning from multiple annotators. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2344–2348.

Ruotian Ma, Tao Gui, Linyang Li, Qi Zhang, Xuanjing Huang, and Yaqian Zhou. 2021. SENT: Sentence-level distant relation extraction via negative training. 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 6201–6213, Online. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Curtis G. Northcutt, Anish Athalye, and Jonas Mueller. 2021. Pervasive label errors in test sets destabilize machine learning benchmarks. *CoRR*, abs/2103.14749.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018. DSGAN: Generative adversarial training for distant supervision relation extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505, Melbourne, Australia. Association for Computational Linguistics.

Borut Sluban, Dragan Gamberger, and Nada Lavrač. 2014. Ensemble-based noise detection: noise ranking and visual performance evaluation. *Data Mining and Knowledge Discovery*, pages 265–303.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training Convolutional Networks with Noisy Labels. *arXiv e-prints*, page arXiv:1406.2080.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826.

Hiroshi Takeda, Soh Yoshida, and Mitsuji Muneyasu. 2021. Training robust deep neural networks on noisy labels using adaptive sample selection with disagreement. *IEEE Access*, pages 141131–141143.

Virginia Wheway. 2001. Using boosting to detect noisy data. In *Advances in Artificial Intelligence. PRICAI 2000 Workshop Reader*, pages 123–130. Springer Berlin Heidelberg.

Yan Yan, Rómer Rosales, Glenn Fung, Ramanathan Subramanian, and Jennifer Dy. 2014. Learning from multiple annotators with varying expertise. pages 291—-327.

Shujian Zhang, Chengyue Gong, and Eunsol Choi. 2021. Learning with different amounts of annotation: From zero to many labels. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7620–7632.

Zhilu Zhang and Mert R. Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 8792–8802, Red Hook, NY, USA. Curran Associates Inc.

Dawei Zhu, Michael A. Hedderich, Fangzhou Zhai, David Ifeoluwa Adelani, and Dietrich Klakow. 2022. Is bert robust to label noise? a study on learning with noisy labels in text classification. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP (Insights at acl), 2022, Dublin, Ireland, May 26*, pages 62–67. Association for Computational Linguistics.

Discourse Relations Classification and Cross-Framework Discourse Relation Classification Through the Lens of Cognitive Dimensions: An Empirical Investigation

Yingxue Fu
School of Computer Science
University of St Andrews
KY16 9SX, UK
yf30@st-andrews.ac.uk

Abstract

Existing discourse formalisms use different taxonomies of discourse relations, which require expert knowledge to understand, posing a challenge for annotation and automatic classification. We show that discourse relations can be effectively captured by some simple cognitively inspired dimensions proposed by Sanders et al. (2018). Our experiments on cross-framework discourse relation classification (PDTB & RST) demonstrate that it is possible to transfer knowledge of discourse relations for one framework to another framework by means of these dimensions, in spite of differences in discourse segmentation of the two frameworks. This manifests the effectiveness of these dimensions in characterizing discourse relations across frameworks. Ablation studies reveal that different dimensions influence different types of discourse relations. The patterns can be explained by the role of dimensions in characterizing and distinguishing different relations. We also report our experimental results on automatic prediction of these dimensions.

1 Introduction

Discourse relations are useful for various downstream NLP tasks, such as text generation (Ji and Huang, 2021) and machine translation (Sim Smith, 2017). However, discourse relations are shaped by multiple sources of information and require expert knowledge for annotation. Since the release of the Penn Discourse Treebank 2.0 (PDTB 2.0) (Prasad et al., 2008), less than 8% improvement has been made in English implicit relation classification in more than ten years (Atwell et al., 2021). Even with the development of contextualized embeddings, this task shows the least improvement in performance compared with other NLP tasks.

Another issue is that existing studies on discourse relation classification are separated into several independent strands of work (Zeldes et al., 2021). The complex nature of discourse gives rise to discourse annotation frameworks which vary in assumptions and definitions of fundamental aspects of discourse, such as what constitutes a discourse relation, what is a basic discourse unit, full-coverage or shallow discourse annotation, and how discourse structure is represented (Fu, 2022).

The leading examples of these annotation frameworks include the Rhetorical Structure Theory (RST) (Mann and Thompson, 1988), the Segmented Discourse Representation Theory (SDRT) (Asher and Lascarides, 2003) and the Discourse Lexicalized Tree-Adjoining Grammar (D-LTAG) (Forbes et al., 2003). These three frameworks have been used in various discourse annotation projects covering different languages. Based on the RST framework, the Rhetorical Structure Theory Discourse Treebank (RST-DT) (Carlson et al., 2001) is developed. SDRT forms the theoretical framework for the ANNODIS corpus (Afantenos et al., 2012), the STAC corpus (Asher et al., 2016) and so on, and D-LTAG is the theoretical foundation for PDTB (Prasad et al., 2008, 2018), which is the largest corpus annotated with discourse relations.

To enable different strands of research to come together and benefit from data across frameworks, we need an interface with which discourse relation classification tasks under different frameworks can be formulated in similar terms, independent of the underlying theoretical assumptions (Zeldes et al., 2021). The UniDim proposal by Sanders et al. (2018) represents one of the influential approaches for this task. The intuition is that discourse relations of different frameworks can be decomposed into cognitive primitives rooted in the Cognitive approach to Coherence Relations (CCR) (Sanders et al., 1992, 1993) (hence denoted as the CCR framework), and people can make use of these elementary notions to relate and compare discourse relations. These primitives are not intended to form a complete and descriptively adequate account of discourse relations but are targeted at a psychologically plausible theory of discourse relations (Sanders et al., 1992). Additional primitives are added in later studies to reach better linguistic and cognitive coverage (Crible and Degand, 2019).

Sanders et al. (2018) and other researchers such as Rehbein et al. (2016) try to test if discourse relations annotated based on the CCR framework are consistently categorized into relations under other frameworks. Their investigation reveals that discrepancies between frameworks arise due to variations in how coherence relations are defined, the methods used to perform the annotation, and the rules governing segmentation, and the alignment of discourse relations is generally many-to-many.

In this study, we aim to assess to what extent these CCR dimensions provide information about discourse relations of different frameworks. We assume that CCR dimensions are annotated in parallel to discourse relation annotations of other frameworks and utilize these dimensions as features in discourse relation classification tasks. The improvement/degradation of performance relative to the case without such features as a measure of the information that these dimensions provide. In this way, we show empirical evidence of the effectiveness of the UniDim proposal in representing and bridging discourse relations of different discourse annotation schemes.

Our contributions include:

- We show that the dimensions of the UniDim proposal effectively capture discourse relations and are useful for training computational systems for discourse relation classification, both for RST relation classification and PDTB explicit and implicit relation classification, yielding significant performance gains. Such elementary cognitive dimensions can be useful features for the challenging task of discourse relation classification.

- We demonstrate that these dimensions can work as an interface for discourse relations across different frameworks. It is possible to train one discourse relation classification model on PDTB and apply the model to the discourse relation classification task in RST with transfer learning and the performance is as high as training a model specifically for RST relation classification, in spite of differences in discourse segmentation between the two frameworks. The CCR dimensions provide an effective means of bridging discourse relations of different frameworks.

- We report experimental results on automatic prediction of these dimensions with RST-DT, PDTB 3.0 and a combination of the two corpora.

2 Related Work

2.1 Mapping Discourse Relations of Different Frameworks

Prior studies on mapping discourse relations of different frameworks adopt varied approaches. Some researchers propose common inventories of relations that are created based on analysis of discourse relations of different frameworks (Benamara and Taboada, 2015; Bunt and Prasad, 2016). Alternatively, an intermediate representation may be used to reduce the number of mappings necessary to harmonize different frameworks (Chiarcos, 2014; Sanders et al., 2018). As there are corpora that contain parallel annotations under different frameworks on the same texts, these corpora are used to identify mappings between discourse relations. Since this approach relies on textual matching, differences in discourse segmentation would hinder relation mapping, leaving only a small number of relations successfully mapped between different frameworks (Bourgonje and Zolotarenko, 2019; Scheffler and Stede, 2016). The study by Demberg et al. (2019) employs the *strong nuclearity hypothesis* (Marcu, 2000) to mitigate this problem. Demberg et al. (2019) show that the Unified Dimension (UniDim) approach is relatively successful in mapping relations between RST-DT and PDTB 2.0.

Roze et al. (2019) investigates the possibility of predicting CCR dimensions automatically. They achieve an accuracy above the baseline of majority class guessing. Furthermore, they try to predict relations of PDTB 2.0 from these dimensions, and it is shown that the accuracy is much lower than that of training a model for predicting PDTB relations directly. The low performance may be attributed to the high level of under-specification in the mapping from PDTB relations to these dimensions and the reverse mapping from dimension combinations to the hierarchical PDTB sense labels, especially when the mapping is not necessarily one-to-one.

Recent studies propose to represent discourse relations as question-answering (QA) pairs (Ko et al., 2022; Pyatkin et al., 2020). While this approach is designed to simplify discourse relation labelling, some relations cannot be expressed by QA pairs (Pyatkin et al., 2020), and evaluation is difficult. Moreover, open-ended QA leads to annotations similar to the GraphBank (Wolf and Gibson, 2004), which has higher complexity than the other frameworks.

2.2 Dimensions in UniDim Proposal

The main approach adopted in the UniDim proposal is to use cognitively inspired dimensions as an intermediate representation and decompose discourse relations of different frameworks into these dimensions so that they can be related and compared. The result contains five dimensions which are rooted in the Cognitive approach to Coherence Relations (CCR) (Sanders et al., 1992, 1993) and some additional dimensions that are added to allow more relations to be better represented (collectively referred to as "UniDim dimensions" or "dimensions of the UniDim proposal" in the following). We give an overview of these dimensions here.

Two segments that may stand in a discourse relation are identified first, the two segments being denoted as S_1 and S_2 in linear order, and the underlying propositions being denoted as P and Q in linear order.

The first dimension is **basic operation**, which has two values: *causal* and *additive*. A causal relation means that the two segments are strongly connected and typically, an implication relation $P \to Q$ can be deduced. In (1), S_2 shows the cause and S_1 gives the consequent. If the two segments are just loosely connected and only a conjunction relation $P \wedge Q$ can be inferred, the value at this dimension is additive, as shown in (2).

(1) [He immigrated to the US,]$_{S_1}$ because [his natural parents were believed to live there.]$_{S_2}$

(2) [She is a painter]$_{S_1}$ and [her studio is a few blocks away.]$_{S_2}$

As indicated in Sanders et al. (2018), basic operation can be used to distinguish causal relations or conditional relations from additive relations or temporal relations.

The second dimension is **source of coherence**. It has two values: semantic and pragmatic in the original proposal (Sanders et al., 1992), later renamed as *objective* and *subjective* in Maat and Sanders (2000), respectively. A relation is objective if the segments are connected because of their propositional content, and the relation holds because the connection is coherent based on world knowledge, as shown in (3). A relation is subjective if the speaker's reasoning or the pragmatic effect of the relation is prominent. (4) shows a claim in S_2 and S_1 is an argument that supports it.

(3) [It was dark outside,]$_{S_1}$ so [he lit up a candle.]$_{S_2}$

(4) [Smoking is unhealthy]$_{S_1}$ and [we should put a limit on it.]$_{S_2}$

This dimension can be used to distinguish relations that are related to real-world situations, such as temporal sequence, and cause-consequence, from argumentative relations, such as claim-argument or evidence-justification (Sanders et al., 2018).

The third dimension is **implication order**. This dimension distinguishes between *non-basic* and *basic* orders of causal relations, and does not apply to additive relations, which are generally symmetric. For a causal relation characterized by $P \to Q$, if S_1 expresses P and S_2 expresses Q (note that S_1 and S_2 are in linear order), then this relation is in basic order, as shown in (6). If S_2 actually expresses P while S_1 expresses Q, this relation is in non-basic order, as shown in (5).

(5) [He did not attend the conference,]$_{S_1}$ because [he received a message telling him not to go.]$_{S_2}$

(6) Because [he received a warning message,]$_{S_1}$ [he did not attend the conference.]$_{S_2}$

It is clear to see that the implication order dimension is mainly used to distinguish relations with directionality, such as cause-result and cause-reason.

The fourth dimension is **polarity**. A relation is characterized by *positive* polarity if the propositions P and Q, expressed by S_1 and S_2, respectively, have the same logical polarity and support each other, as shown in (7). A relation is of *negative* polarity if the relation involves the juxtaposition of $\neg P$ and P or $\neg Q$ and Q in the two segments, as shown in (8). In this example, a positive polarity would require a reason or result that supports the decision of closing the library.

(7) [We like the garden]$_{S_1}$ because [it is pretty.]$_{S_2}$

(8) [The university library was closed]$_{S_1}$ although [students wanted more space for study.]$_{S_2}$

This dimension is useful for capturing contrastive, adversative and concession relations (Sanders et al., 2018).

The fifth dimension is **temporality**, which distinguishes between *temporal* and *non-temporal* relations. Under temporal relations, temporality has three values: *synchronous*, *chronological* and *anti-chronological*. Synchronous relations are those temporal relations which feature simultaneous occurrence of events. If events described in the segments happen in temporal order, then the relation is chronological, otherwise the relation is anti-chronological.

In order to characterize more relations, additional dimensions are introduced, including **specificity**, **lists** and **alternatives** for additive relations, and **conditionals** and **goal-oriented relations** for causal relations (denoted collectively as "additional dimensions" in the following).

3 Methodology

Since RST-DT and PDTB both use the WSJ articles of the Penn Treebank, cross-framework relation classification of RST and PDTB by automatic means would be less influenced by domain shift. Therefore, we focus on the two frameworks. For PDTB, we use PDTB 3.0, which is newer and introduced systematic changes.

As we are primarily interested in the effectiveness of UniDim dimensions rather than improving algorithms for discourse relation classification, simple models are implemented in the experiments.

3.1 Discourse Relation Classification

Discourse relation classification is a typical multi-class classification task. Given a span/argument pair with tokens $S = [CLS]$, $S_1^{(1)}$ ... $S_m^{(1)}$, $[SEP]$, $S_1^{(2)}$ ... $S_n^{(2)}$, we obtain the representation of the sequence from a pre-trained language model, denoted as $f_{PLM}(S)$, and the embeddings of the dimensions E are obtained from embedding layers, where the embeddings are initialized from uniform distributions and trainable. The representation of the input and the embeddings of dimensions are concatenated:

$$h_S = f_{PLM}(S) \oplus Edim_{pol} \oplus Edim_{bop} \oplus ... \quad (1)$$

The dim_{pol} and dim_{bop} ... represents the UniDim dimensions, including polarity, basic operation, implication order, source of coherence, temporaltiy, specificity, alternative, conditional and goal.

The representation is fed to two two-layer feed-forward networks (FFNs) with LeakyReLU as activation functions:

$$\hat{h} = g_2(W_2 * g_1(W_1 * h_S)) \quad (2)$$

where g_1 and g_2 represent the non-linear activation functions of first and the second FFNs, respectively. W_1 and W_2 denote weights of the first layers of the two FFNs, and bias terms are omitted for clarity.

A classifier layer is configured on top of the second FFN. The predicted result $\hat{y}$ is obtained with:

$$\hat{y} = softmax(W_3 * \hat{h}) \quad (3)$$

Cross-entropy loss is used in the loss function:

$$\mathcal{L}_c = -\sum_{i=1}^{N}\sum_{l=1}^{C} c_l^i \log p(c_l^i) \quad (4)$$

where N is the batch size, C is the total number of classes, and $p(c_l^i)$ is the probability predicted for a class c.

In this design, we take our experiments with transfer learning for cross-framework discourse relation classification into consideration, as we try to keep the architecture and only replace the last classifier layer to fit the model on new data. Moreover, our preliminary experiments indicate that removing the second FFN causes a significant performance drop.

Baseline model The BertForSequenceClassification model from the Transformers library (Wolf et al., 2020) is used as the baseline model, in which a classifier layer is added on top of the contextualized embeddings of the input sequence. For an input sequence S, its representation is obtained with:

$$h_S = f_{PLM}(S) \quad (5)$$

The predicted result $\hat{y}$ is obtained with:

$$\hat{y} = softmax(W_b * h_S) \quad (6)$$

As shown in Kim et al. (2020), this model is a strong baseline. We use the *bert-base-uncased* BERT model in all our experiments for comparison of experimental results.

3.2 Cross-framework Discourse Relation Classification

We hypothesize that if UniDim dimensions form an effective "interlingua" of discourse relations from different frameworks, we can train a model for discourse relation classification in one framework and apply the model for relation classification in another framework without much modification. The transfer learning framework can be used for this experiment.

As PDTB 3.0 is much larger than RST-DT, a natural choice would be to treat PDTB relation classification as the source task and RST relation classification as the target task (Wang et al., 2019).

We first train a model as described in section 3.1 on all the PDTB data, and freeze all the layers but the last classifier layer so that the model can be fit on RST data.

Formally, for a pair of PDTB arguments $P = [CLS], A_1^{(1)} ... A_m^{(1)}, [SEP], A_1^{(2)} ... A_n^{(2)}$, we obtain the representation of sequence P with equation (1). Through training, the parameters in equation (2) are learnt for the PDTB relation classification task. With these parameters, for an RST span pair $R = [CLS], R_1^{(1)}, ..., R_m^{(1)}, [SEP], R_1^{(2)}, ..., R_n^{(2)}$, we first obtain the representation of sequence R with equation (1), denoted as h_R, and with the parameters learnt for PDTB relation classification, we obtain the representation $\widehat{h_R}$:

$$\widehat{h_R} = g_2(W_2 * g_1(W_1 * h_R)) \tag{7}$$

The predicted result $\hat{y}$ for RST relation classification is obtained with:

$$\hat{y} = softmax(W_r * \widehat{h_R}) \tag{8}$$

where W_r is the weight to be learnt for RST relation classification.

Baseline model As we transfer knowledge from PDTB relation classification to RST relation classification, the baseline model is a model trained specifically for RST relation classification with BERT embeddings and UniDim dimensions as input. For the baseline model in section 3.1, where only BERT embeddings are used, we train a model for PDTB relation classification and apply the model to RST relation classification without using UniDim dimensions.

3.3 Automatic UniDim Dimension Prediction

Since the dimensions may be related to each other, we train one model for predicting the nine dimensions in equation 1 together.

For an input sequence S, we obtain its representation h_S with equation 5. A two-layer FFN f with LeakyReLU activation function is applied to h_S before nine classification layers $c_{i|i=1...9}$ are applied:

$$\hat{y} = softmax(W_{c_i} * f(h_S)) \tag{9}$$

We train the model on PDTB, RST and the combination of PDTB and RST data, respectively. The results reported in Roze et al. (2019) are our baseline.

4 Experiments

We use the mapping table given in Sanders et al. (2018) (Appendix A) for obtaining the dimension values for relation labels of RST-DT. As no mapping table is provided for PDTB 3.0, we create the mapping table by ourselves (Appendix B).

4.1 Data Preprocessing

We binarize the RST trees based on the procedure in Ji and Eisenstein (2014) and extract pairs of spans that are connected by a relation. Following Sanders et al. (2018), we exclude *Same-Unit* and *Attribution* relations from RST-DT, leaving 16 relations. We use the standard split of the corpus and take 20% from the training set for validation.

Since PDTB level-2 relations carry specific and generally more useful information, we focus on level-2 relation classification for PDTB. We exclude relations that have fewer than 100 instances to alleviate data imbalance, as suggested in Kim et al. (2020). We follow the data split in Ji and Eisenstein (2015), using sections 2-20 for training, 0-1 for validation and 21-22 for testing.

We use the pre-trained BERT model (Devlin et al., 2019) for obtaining contextualized embeddings and the *[CLS]* and *[SEP]* tokens are inserted following the settings of the BERT model, which is shown to benefit inter-sentential (Shi and Demberg, 2019) and intra-sentential (Zhao and Webber, 2021) implicit discourse relation classification.

Among the UniDim dimensions, we exclude *list* because this dimension is proposed for representing the *List* relation in PDTB, which has been removed from the sense hierarchy in PDTB 3.0. Following Roze et al. (2019), we merge *specificity-example* and *specificity-equivalence* into specificity, and add the *NS* label in cases of ambiguity or under-specification. The *N.A.* label is kept when it appears on its own to reflect the fact that some dimensions do not apply to certain types of relations. The default values of additional dimensions are set to negative because they are only applicable to some relations and typically have binary values.

On the whole, the dimensions are heavily imbalanced and have high degree of under-specification. Statistics for the distribution of these dimensions are shown in Appendix C. Hyper-parameter settings and model training details are described in

Appendix D.

4.2 Evaluation

For RST relation classification, the settings of the DISRPT 2021 shared task on relation classification (Zeldes et al., 2021) are the closest to ours. We report their best accuracy on RST-DT (Gessler et al., 2021) alongside our baseline model results for comparison.

After preprocessing, we perform 12-way explicit relation classification and 14-way implicit relation classification for PDTB. While most of the previous studies use PDTB 2.0 and recent studies on PDTB 3.0 only focus on implicit relation classification, when settings of previous studies are close to ours, we report their results alongside our baseline results[1].

4.3 Results and Discussion

We report our experimental results on the test sets, which are computed with the Scikit-Learn library (Pedregosa et al., 2011). We can expect that RST and PDTB data show different patterns. For RST, the dimension values for end labels may be clear, but when end labels are grouped into a class, the values could be rather mixed. For PDTB, as L2 sense classification is performed, the process of grouping relations into broader classes happens at L3, which only encodes directionality, and dimensions that are related to directionality are affected, such as implication order, but the other dimensions are not influenced. Therefore, dimension values for PDTB classes tend to be less ambiguous. Moreover, data amount differences are likely to have notable influence on the results. We do not report the results of additional dimensions separately because their individual effects are not obvious.

4.3.1 RST Relation Classification

Table 1 shows results on RST-DT. When UniDim dimensions are added as features, a significant performance gain can be obtained. Some relations can be recognized with 100% accuracy. However, relations including *Comparison*, *Manner-means*, *Summary* and *Textual-Organization* cannot be recognized. From Fig. 3 in Appendix E, it is clear that these relations have small amounts of training data. As we focus on broader classes rather than end labels in relation classification, we can see from the mapping table in Appendix A that dimension values under these classes are mixed. It is difficult for the model to learn patterns from the data.

	P	R	$F1$	$P_{b.}$	$R_{b.}$	$F1_{b.}$	$C.$
Background	1.00	1.00	1.00	0.47	0.35	0.40	111
Cause	0.92	0.70	0.79	0.50	0.17	0.25	82
Comparison	0.00	0.00	0.00	0.61	0.38	0.47	29
Condition	1.00	1.00	1.00	0.79	0.71	0.75	48
Contrast	0.99	1.00	0.99	0.75	0.68	0.72	146
Elaboration	0.75	1.00	0.86	0.65	0.88	0.75	796
Enablement	0.92	1.00	0.96	0.61	0.85	0.71	46
Evaluation	0.99	1.00	0.99	0.29	0.14	0.19	80
Explanation	0.72	0.97	0.83	0.46	0.27	0.34	110
Joint	1.00	0.03	0.06	0.67	0.62	0.64	212
Manner-Means	0.00	0.00	0.00	0.68	0.48	0.57	27
Summary	0.00	0.00	0.00	0.88	0.47	0.61	32
Temporal	1.00	1.00	1.00	0.74	0.27	0.40	73
Textual-Organization	0.00	0.00	0.00	0.44	0.44	0.44	9
Topic-Change	0.28	1.00	0.44	0.28	0.38	0.32	13
Topic-Comment	0.71	0.21	0.32	0.00	0.00	0.00	24
Acc.	**0.81**			0.63 (vs DISRPT 2021: 0.67)			
Macro-F1	**0.64**	**0.62**	**0.58**	0.55	0.44	0.47	1838

Table 1: Results of RST relation classification. The columns in blue show the results of our method and uncolored columns show the results of the baseline model, and the last column shows the count of occurrences of each relation in the test set. We use this convention in reporting the results.

To have a better understanding of the influence of each dimension on the results, we performed ablation studies and the results are shown in Table 2.

	Acc	P	R	$F1$
Total	0.81	0.64	0.62	0.58
-Pol.	0.74	0.49	0.48	**0.48**
-Basic Op.	0.78	0.52	0.58	0.53
-SoC.	0.78	0.52	0.58	0.53
-Impl. order	0.81	0.58	0.60	0.55
-Temp.	0.80	0.59	0.60	0.55
-Add.	0.80	0.52	0.59	0.54

Table 2: Results of ablation studies for RST relation classification, showing the overall accuracy (Acc), precision (P), recall(R) and macro-averaged F1 ($F1$) for dimensions of polarity (Pol.), basic operation (Basic Op.), source of coherence (SoC.), implication order (Impl. order), temporality (Temp.) and additional dimensions (Add.), respectively.

As shown in Table 2, removing the polarity dimension causes the biggest performance drop in macro-averaged F1. By examining the detailed results (Table 33, Appendix L), we find that removing this dimension has noticeable influence on the recognition of *Contrast*($\downarrow$ 0.41), *Evaluation*($\downarrow$ 0.26), *Topic-Change*($\downarrow$ 0.44) and *Topic-Comment*($\downarrow$ 0.32). The correlation between *Contrast* and this dimension is self-evident. Examination of the mapping table suggests that the rest of these relations have ambiguous or mixed values in the other dimensions and their data amounts are small, making it difficult for the model to learn any patterns.

4.3.2 PDTB Explicit Relation Classification

Table 3 shows the results of 12-way explicit relation classification. The overall accuracy score is high and the majority of the relations can be recognized with near perfect performance, which means that the UniDim dimensions are effective

[1] We build and run all the baseline models mentioned in section 3.1 and section 3.2 by ourselves.

in characterizing most of the PDTB explicit relations. However, in spite of the noticeable improvement in overall accuracy, our method does not show improvement over the baseline model in macro-averaged F1 score. This is likely due to the strong reliance of pre-trained language models on lexical cues in discourse relation classification tasks (Kim et al., 2020) and these lexical cues are effective features for this task. Moreover, with our approach, the *Level-of-detail* and *Substitution* relations cannot be recognized. The two relations have the smallest data amount, and in terms of dimension values, *Substitution* is similar to *Concession* and *Level-of-detail* is similar to *Manner*. It is possible that the model predicts *Manner* for instances of *Level-of-detail*, which explains the lower precision for *Manner*.

	P	R	$F1$	$P_{b.}$	$R_{b.}$	$F1_{b.}$	$C.$
Asynchronous	1.00	1.00	1.00	0.97	0.87	0.92	127
Cause	1.00	1.00	1.00	0.82	0.89	0.85	115
Concession	0.96	1.00	0.98	0.89	0.95	0.92	285
Condition	1.00	1.00	1.00	0.93	0.92	0.93	61
Conjunction	1.00	1.00	1.00	0.97	0.96	0.96	516
Contrast	1.00	1.00	1.00	0.52	0.48	0.50	50
Disjunction	1.00	1.00	1.00	0.90	1.00	0.95	18
Level-of-detail	0.00	0.00	0.00	0.71	0.75	0.73	20
Manner	0.35	1.00	0.52	0.42	0.91	0.57	11
Purpose	1.00	1.00	1.00	0.62	0.45	0.52	29
Substitution	0.00	0.00	0.00	1.00	0.92	0.96	13
Synchronous	1.00	1.00	1.00	0.81	0.71	0.76	126
Acc.	**0.98**			0.89			
Macro-F1	**0.78**	**0.83**	**0.79**	0.80	0.82	0.80	1371

Table 3: Results of PDTB explicit relation classification.

The results of ablation studies are shown in Table 4. Removing the source of coherence dimension causes the biggest performance drop in macro-averaged F1. Through examining the detailed results, we find that without this dimension, the *Disjunction* relation cannot be recognized. Meanwhile, removing this dimension causes a drop of 0.15 for identifying the *Contrast* relation and a drop of 0.14 for recognizing the *Synchronous* relation. The *Disjunction* relation has a small data amount, and the model might predict *Contrast* for instances of *Disjunction*, since they are similar in the absence of this dimension, which may account for the lower precision for *Contrast*.

	Acc	P	R	$F1$
Total	0.98	0.78	0.83	0.79
-Pol.	0.95	0.74	0.81	0.76
-Basic Op.	0.98	0.78	0.83	0.79
-SoC.	0.94	0.67	0.73	**0.68**
-Impl. order	0.98	0.78	0.83	0.79
-Temp.	0.95	0.76	0.81	0.77
-Add.	0.96	0.73	0.73	0.73

Table 4: Results of ablation studies for PDTB explicit relation classification.

4.3.3 PDTB Implicit Relation Classification

Table 5 shows the results of 14-way implicit relation classification. The previous best result under similar settings is 0.64 in overall accuracy (Kim et al., 2020), which is achieved with *large-cased* XLNet (Yang et al., 2019). Our baseline 56% accuracy is consistent with the results in Kim et al. (2020).

	P	R	$F1$	$P_{b.}$	$R_{b.}$	$F1_{b.}$	$C.$
Asynchronous	1.00	1.00	1.00	0.62	0.61	0.62	95
Cause	1.00	1.00	1.00	0.60	0.63	0.61	366
Cause+Belief	1.00	0.42	0.59	0.00	0.00	0.00	12
Concession	1.00	0.92	0.96	0.44	0.40	0.42	84
Condition	1.00	1.00	1.00	0.71	0.42	0.53	12
Conjunction	0.90	1.00	0.95	0.49	0.61	0.54	221
Contrast	0.98	1.00	0.99	0.45	0.42	0.43	50
Equivalence	0.00	0.00	0.00	0.12	0.04	0.06	24
Instantiation	0.00	0.00	0.00	0.77	0.54	0.64	107
Level-of-detail	0.60	1.00	0.75	0.45	0.48	0.46	180
Manner	0.00	0.00	0.00	0.38	0.60	0.46	15
Purpose	0.92	0.94	0.93	0.92	0.98	0.95	88
Substitution	0.75	1.00	0.86	0.43	0.48	0.45	21
Synchronous	0.87	0.97	0.92	0.27	0.10	0.15	40
Acc.	**0.87**			0.56			
Macro-F1	**0.72**	**0.73**	**0.71**	0.48	0.45	0.45	1315

Table 5: Results of PDTB implicit relation classification.

As is shown in Table 5, adding UniDim dimensions brings significant performance gain for this task, which is challenging for the baseline model. Meanwhile, we notice that relations including *Equivalence*, *Instantiation* and *Manner* are difficult to recognize. In terms of dimension values, *Equivalence* is similar to *Conjunction*, which has a much larger amount of data. It is likely that the model predicts *Conjunction* for *Equivalence*, hence the lower precision for *Conjunction*. *Instantiation*, *Manner* and *Level-of-detail* have the same dimension values, and as the data amount for *Level-of-detail* is much larger, the model may predict *Level-of-detail* for instances of the other two relations, causing the precision score for *Level-of-detail* to go down.

The results of ablation studies are shown in Table 6. Both the implication order dimension and the additional dimensions have substantial influence on the F1 score. Removing the implication order dimension does not cause much decrease in the overall accuracy score but mainly lowers the F1 score, while removing the additional dimensions reduces both the overall accuracy score and the F1 score.

	Acc	P	R	$F1$
Total	0.87	0.72	0.73	0.71
-Pol.	0.87	0.71	0.71	0.70
-Basic Op.	0.87	0.72	0.73	0.71
-SoC.	0.87	0.72	0.73	0.71
-Impl. order	0.86	0.57	0.64	**0.60**
-Temp.	0.87	0.72	0.73	0.71
-Add.	0.73	0.64	0.64	**0.62**

Table 6: Results of ablation studies for PDTB implicit relation classification.

Detailed results (Table 26 in Appendix J) show that removing the implication order dimension causes a drop of 0.07 in recognizing *Concession*, a drop of 0.86 in recognizing *Substitution* and a drop of 0.59 in recognizing *Cause+Belief*. As the last two relations cannot be recognized, the macro-averaged F1 shows a significant decrease. Similarly, this is associated with differences in data amount and how different relations can be distinguished from each other without the dimension, for instance, *Substitution* has a small data amount, and without the implication order dimension, the model might confuse this relation with *Concession* and predict *Concession* for instances of both relations, which may explain the lower precision for *Concession*. If the additional dimensions are removed, major relations that are impacted include *Condition*(↓ 0.14), *Conjunction*(↓ 0.37), and *Level-of-detail*(↓ 0.75). In this case, the *Level-of-detail* relation cannot be identified. Without this dimension, *Level-of-detail* has the same dimension values as *Conjunction*, which has a larger data amount. The model may predict *Conjunction* for both classes, which causes precision for *Conjunction* to decrease.

4.3.4 Cross-Framework Discourse Relation Classification

As RST does not distinguish explicit and implicit relations, we train a model on the whole PDTB data for the source task. We show the overall performance of transfer learning from PDTB to RST in Table 7. The settings of the DISRPT 2021 shared task are the closest to our experiments, and their best results (Gessler et al., 2021) are shown alongside the baseline model for comparison. As is clear from the table, the results of transfer learning based on the baseline BERT model show noticeable effect of negative transfer (0.63 → 0.58 in overall accuracy and 0.47 → 0.33 in F1 score), while with our method, the overall accuracy does not show any decrease and the F1 score is only 1% lower. This shows that the UniDim dimensions may serve as an effective interface for relations of different frameworks. The detailed results for the source and target tasks are shown in Tables 39 and 40 in Appendix M.

Task	Acc.	Macro-F1
target RST (BERT+Dim)	**0.81**	**0.57**
RST-specific (BERT+Dim) from Table 1	0.81	0.58
src PDTB total (BERT+Dim)	0.86	0.67
target RST (BERT only)	0.58	0.33
RST-specific (BERT only) from Table 1	0.63	0.47
src PDTB total (BERT only)	0.71 (vs. DISRPT 2021: 0.74)	0.61

Table 7: Results of transfer learning from PDTB to RST.

4.3.5 Automatic Dimension Prediction

We show our experimental results of automatic prediction of UniDim dimensions in Table 8. As is clear from the table, reasonable performance for this task can be achieved. Note that the baseline results are based on PDTB 2.0 and separate classifiers are trained for each dimension.

The performance on PDTB is higher than on RST data with the exception of *Temporality* and *Goal*. As PDTB allows multi-sense annotation, instances labeled with temporal relations might be annotated with labels of causal relations, and instances for which a *Purpose* relation can be inferred (captured by the *Goal* dimension), a *Manner* relation is also possible (not involving the *Goal* dimension), which poses a challenge for machine learning systems.

Moreover, combining the two corpora to augment training data does not improve the performance over using PDTB data alone but it is helpful for improving performance on RST data. RST data amount is much smaller and adding more data is beneficial. As relations of the two frameworks may not be completely compatible and combining the two corpora might introduce inconsistent and redundant data, combining the datasets is likely to be more useful in low-resource settings.

	PDTB		RST		PDTB+RST		Baseline	
	Acc.	Macro-F1	Acc.	Macro-F1	Acc.	Macro-F1	Acc.	Macro-F1
Pol.	0.92	0.57	0.85	0.58	0.89	0.56	0.82	0.50
Basic Op.	0.80	0.52	0.76	0.45	0.77	0.50	0.76	0.38
SoC.	0.75	0.72	0.67	0.45	0.70	0.59	0.68	0.50
Impl. order	0.76	0.50	0.75	0.38	0.75	0.48	0.78	0.41
Temp.	0.79	0.59	0.86	0.30	0.82	0.43	0.73	0.48
Spec.	0.87	0.65	0.80	0.72	0.83	0.66	0.85	-
Alter.	1.00	0.95	1.00	0.50	1.00	0.95	0.99	-
Cond.	0.99	0.86	0.98	0.83	0.98	0.83	0.99	-
Goal	0.91	0.75	0.97	0.75	0.93	0.74	-	-

Table 8: Results of UniDim dimension prediction. Blue columns show classification accuracy and grey columns show macro-averaged F1.

5 Conclusion and Future Work

By incorporating the UniDim dimensions proposed in Sanders et al. (2018) in discourse relation classification tasks, we obtain quantitative results of the effectiveness of these dimensions in capturing discourse relations of different frameworks and bridging discourse relations across frameworks. Ablation studies reveal the influence of these dimensions on different types of discourse relations. Meanwhile, we show that these dimensions can be predicted automatically with a simple model. These dimensions are potentially useful features for discourse relation classification across frameworks. Therefore, in future work, we plan to incorporate automatically predicted dimensions in our models.

6 Limitations

Since we need to create the mapping table for PDTB 3.0 by ourselves, it is unavoidable that there may be errors and inconsistencies with existing mapping tables for the other frameworks.

Meanwhile, in the mapping table provided in Sanders et al. (2018), to obtain the values of the dimensions, we need all the information of a relation label, for instance, to represent an RST relation label with dimensions, we need the nuclearity label and whether the relation is mono-nuclear or multi-nuclear in addition to the relation label itself, and in the case of a PDTB relation, we need the relation label and the order of the arguments. This is because these dimensions are not incorporated in the annotation process of RST-DT and PDTB, and only a general mapping is possible. We consider the resultant ambiguity and under-specification unavoidable.

7 Ethics Statement

This study does not involve special ethical considerations. The potential impact may include providing computational evidence of the validity of cognitive study of discourse relations and attracting attention to cognitive frameworks of discourse, which may spur fine-grained research on the correlation between cognitive dimensions and different discourse relations and how different language models perform from this perspective.

References

Stergos Afantenos, Nicholas Asher, Farah Benamara, Myriam Bras, Cécile Fabre, Mai Ho-dac, Anne Le Draoulec, Philippe Muller, Marie-Paule Péry-Woodley, Laurent Prévot, Josette Rebeyrolles, Ludovic Tanguy, Marianne Vergez-Couret, and Laure Vieu. 2012. An empirical resource for discovering cognitive principles of discourse organisation: the ANNODIS corpus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2727–2734, Istanbul, Turkey. European Language Resources Association (ELRA).

Nicholas Asher, Julie Hunter, Mathieu Morey, Benamara Farah, and Stergos Afantenos. 2016. Discourse structure and dialogue acts in multiparty dialogue: the STAC corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2721–2727, Portorož, Slovenia. European Language Resources Association (ELRA).

Nicholas Asher and Alex Lascarides. 2003. *Logics of Conversation*. Cambridge University Press, United States.

Katherine Atwell, Junyi Jessy Li, and Malihe Alikhani. 2021. Where are we in discourse relation recognition? In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 314–325, Singapore and Online. Association for Computational Linguistics.

Farah Benamara and Maite Taboada. 2015. Mapping different rhetorical relation annotations: A proposal. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 147–152, Denver, Colorado. Association for Computational Linguistics.

Peter Bourgonje and Olha Zolotarenko. 2019. Toward cross-theory discourse relation annotation. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pages 7–11, Minneapolis, MN. Association for Computational Linguistics.

Harry Bunt and Rashmi Prasad. 2016. ISO DR-Core (ISO 24617-8): Core concepts for the annotation of discourse relations. In *Proceedings 12th Joint ACL-ISO Workshop on Interoperable Semantic Annotation (ISA-12)*, pages 45–54.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurovsky. 2001. Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. In *Proceedings of the Second SIGdial Workshop on Discourse and Dialogue*.

Christian Chiarcos. 2014. Towards interoperable discourse annotation. discourse features in the ontologies of linguistic annotation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4569–4577, Reykjavik, Iceland. European Language Resources Association (ELRA).

Ludivine Crible and Liesbeth Degand. 2019. Domains and functions: A two-dimensional account of discourse markers. *Discours. Revue de linguistique, psycholinguistique et informatique. A journal of linguistics, psycholinguistics and computational linguistics*, (24).

Vera Demberg, Merel CJ Scholman, and Fatemeh Torabi Asr. 2019. How compatible are our discourse annotation frameworks? insights from mapping rst-dt and pdtb annotations. *Dialogue & Discourse*, 10(1):87–135.

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.

Katherine Forbes, Eleni Miltsakaki, Rashmi Prasad, Anoop Sarkar, Aravind Joshi, and Bonnie Webber. 2003. D-LTAG system: Discourse parsing with a lexicalized tree-adjoining grammar. *Journal of Logic, Language and Information*, 12(3):261–279.

Yingxue Fu. 2022. Towards unification of discourse annotation frameworks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 132–142, Dublin, Ireland. Association for Computational Linguistics.

Luke Gessler, Shabnam Behzad, Yang Janet Liu, Siyao Peng, Yilun Zhu, and Amir Zeldes. 2021. DisCoDisCo at the DISRPT2021 shared task: A system for discourse segmentation, classification, and connective detection. In *Proceedings of the 2nd Shared Task on Discourse Relation Parsing and Treebanking (DISRPT 2021)*, pages 51–62, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Haozhe Ji and Minlie Huang. 2021. DiscoDVT: Generating long text with discourse-aware discrete variational transformer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4224, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24, Baltimore, Maryland. Association for Computational Linguistics.

Yangfeng Ji and Jacob Eisenstein. 2015. One vector is not enough: Entity-augmented distributed semantics for discourse relations. *Transactions of the Association for Computational Linguistics*, 3:329–344.

Najoung Kim, Song Feng, Chulaka Gunasekara, and Luis Lastras. 2020. Implicit discourse relation classification: We need to talk about evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5404–5414, Online. Association for Computational Linguistics.

Wei-Jen Ko, Cutter Dalton, Mark Simmons, Eliza Fisher, Greg Durrett, and Junyi Jessy Li. 2022. Discourse comprehension: A question answering framework to represent sentence connections. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11752–11764, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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

Henk Pander Maat and Ted Sanders. 2000. Domains of use or subjectivity? the distribution of three dutch causal connectives explained. *Topics in English Linguistics*, 33:57–82.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Daniel Marcu. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. MIT press.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32.

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(85):2825–2830.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Rashmi Prasad, Bonnie Webber, and Alan Lee. 2018. Discourse annotation in the PDTB: The next generation. In *Proceedings 14th Joint ACL - ISO Workshop on Interoperable Semantic Annotation*, pages 87–97, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Valentina Pyatkin, Ayal Klein, Reut Tsarfaty, and Ido Dagan. 2020. QADiscourse - Discourse Relations as QA Pairs: Representation, Crowdsourcing and Baselines. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2804–2819, Online. Association for Computational Linguistics.

Ines Rehbein, Merel Scholman, and Vera Demberg. 2016. Annotating discourse relations in spoken language: A comparison of the PDTB and CCR frameworks. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1039–1046, Portorož, Slovenia. European Language Resources Association (ELRA).

Charlotte Roze, Chloé Braud, and Philippe Muller. 2019. Which aspects of discourse relations are hard to learn? primitive decomposition for discourse relation classification. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 432–441, Stockholm, Sweden. Association for Computational Linguistics.

Ted J. M. Sanders, Wilbert P. M. Spooren, and Leo G. M. Noordman. 1993. Coherence relations in a cognitive theory of discourse representation. *Cognitive Linguistics*, 4(2):93–134.

Ted JM Sanders, Vera Demberg, Jet Hoek, Merel CJ Scholman, Fatemeh Torabi Asr, Sandrine Zufferey, and Jacqueline Evers-Vermeul. 2018. Unifying dimensions in coherence relations: How various annotation frameworks are related. *Corpus Linguistics and Linguistic Theory*.

Ted JM Sanders, Wilbert PM Spooren, and Leo GM Noordman. 1992. Toward a taxonomy of coherence relations. *Discourse processes*, 15(1):1–35.

Tatjana Scheffler and Manfred Stede. 2016. Mapping PDTB-style connective annotation to RST-style discourse annotation. In *Proceedings of the 13th Conference on Natural Language Processing*, pages 242–247.

Wei Shi and Vera Demberg. 2019. Next sentence prediction helps implicit discourse relation classification within and across domains. 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 5790–5796, Hong Kong, China. Association for Computational Linguistics.

Karin Sim Smith. 2017. On integrating discourse in machine translation. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 110–121, Copenhagen, Denmark. Association for Computational Linguistics.

Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. 2019. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11293–11302.

Florian Wolf and Edward Gibson. 2004. Representing discourse coherence: A corpus-based analysis. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 134–140, Geneva, Switzerland. COLING.

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

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Amir Zeldes, Yang Janet Liu, Mikel Iruskieta, Philippe Muller, Chloé Braud, and Sonia Badene. 2021. The DISRPT 2021 shared task on elementary discourse unit segmentation, connective detection, and relation classification. In *Proceedings of the 2nd Shared Task on Discourse Relation Parsing and Treebanking (DISRPT 2021)*, pages 1–12, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zheng Zhao and Bonnie Webber. 2021. Revisiting shallow discourse parsing in the PDTB-3: Handling intrasentential implicits. In *Proceedings of the 2nd Workshop on Computational Approaches to Discourse*, pages 107–121, Punta Cana, Dominican Republic and Online. Association for Computational Linguistics.

A RST to UniDim Dimension Mapping Table

Table 9 shows the mapping of RST-DT relation labels to UniDim dimensions.

Class	End label	Nuc.	N-S	Pol.	Basic Op.	Impl. order	SoC	Temp.	Add. features
Background	Background	Mono	N-S	pos/neg	add	N.A.	obj	anti/N.A.	
	Background	Mono	S-N	pos/neg	add	N.A.	obj	chron/N.A.	
	Circumstance	Mono		pos/neg	add	N.A.	obj	syn/N.A.	
Cause	Cause	Mono	N-S	pos	cau	bas	obj	chron	
	Cause	Mono	S-N	pos	cau	non-b	obj	anti	
	Cause-result	Multi		pos	cau	bas/non-b	obj	chron/anti	
	Result	Mono	N-S	pos	cau	non-b	obj	anti	
	Result	Mono	S-N	pos	cau	bas	obj	chron	
	Consequence-n	Mono	N-S	pos	cau	non-b	obj	anti	
	Consequence-n	Mono	S-N	pos	cau	bas	obj	chron	
	Consequence-s	Mono	N-S	pos	cau	bas	obj	chron	
	Consequence-s	Mono	S-N	pos	cau	non-b	obj	anti	
	Consequence	Multi		pos	cau	bas/non-b	obj	chron/anti	
Comparison	Comparison	Both		pos	add	N.A.	obj/sub	N.A.	
	Preference	Mono		neg	add	N.A.	obj/sub	N.A.	
	Analogy	Both		pos	add	N.A.	sub	N.A.	
	Proportion	Multi		pos	add/cau	any	obj/sub	any	
Conditional	Condition	Mono	N-S	pos/neg	cau	non-b	obj/sub	anti/N.A.	conditional
	Condition	Mono	S-N	pos/neg	cau	bas	obj/sub	chron/N.A.	conditional
	Hypothetical	Mono	N-S	pos	cau	non-b	sub	N.A.	conditional
	Hypothetical	Mono	S-N	pos	cau	bas	sub	N.A.	conditional
	Contingency	Mono	N-S	pos/neg	cau	non-b	obj	anti	conditional
	Contingency	Mono	S-N	pos/neg	cau	bas	obj	chron	conditional
	Otherwise	Mono	N-S	neg	cau	bas	obj/sub	chron/N.A.	conditional
	Otherwise	Multi		neg	cau	bas	obj/sub	chron/N.A.	conditional
Contrast	Contrast	Multi		neg	add	N.A.	obj/sub	any	
	Concession	Mono	N-S	neg	cau	non-b	obj/sub	anti/N.A.	
	Concession	Mono	S-N	neg	cau	bas	obj/sub	chron/N.A.	
	Antithesis	Mono		neg	add/cau	any	obj/sub	any	
Elaboration	El.-additional	Mono		pos	add	N.A.	obj/sub	N.A.	
	El.-gen.-spec.	Mono		pos	add	N.A.	obj/sub	N.A.	specificity
	El.-part-whole	Mono		pos	add	N.A.	obj	N.A.	specificity
	El.-process-step	Mono		pos	add	N.A.	obj	N.A.	specificity
	El.-object-attr.	Mono		pos	add	N.A.	obj	N.A.	specificity
	El.-set-member	Mono		pos	add	N.A.	obj	N.A.	spec.-ex.
	Example	Mono		pos	add	N.A.	obj	N.A.	spec.-ex.
	Definition	Mono		pos	add	N.A.	obj	N.A.	specificity
Enablement	Purpose	Mono	N-S	pos	cau	bas	obj/sub	chron/N.A.	goal
	Purpose	Mono	S-N	pos	cau	non-b	obj/sub	anti/N.A.	goal
	Enablement	Mono	N-S	pos	cau	non-b	obj/sub	anti/N.A.	goal
	Enablement	Mono	S-N	pos	cau	bas	obj/sub	chron/N.A.	goal
Evaluation	Evaluation	Both		pos	add/cau	any	sub	N.A.	specificity
	Interpretation	Both		pos	add/cau	any	sub	N.A.	specificity
	Conclusion	Mono	N-S	pos	cau	bas	sub	N.A.	specificity
	Conclusion	Mono	S-N	pos	cau	non-b	sub	N.A.	specificity
	Conclusion	Multi		pos	cau	bas/non-b	sub	N.A.	specificity
	Comment	Mono		pos	add	N.A.	sub	N.A.	specificity
Explanation	Evidence	Mono	N-S	pos	cau	non-b	sub	anti	
	Evidence	Mono	S-N	pos	cau	bas	sub	chron	
	Exp.-argument.	Mono	N-S	pos	cau	non-b	obj	anti	
	Exp.-argument.	Mono	S-N	pos	cau	bas	obj	chron	
	Reason	Mono	N-S	pos	cau	non-b	obj	anti	
	Reason	Mono	S-N	pos	cau	bas	obj	chron	
	Reason	Multi		pos	cau	bas/non-b	obj	chron/anti	
Joint	List	Multi		pos	add	N.A.	obj/sub	syn/chron/N.A.	list
	Disjunction	Multi		pos/neg	add	N.A.	obj/sub	syn/N.A.	alternative
Summary	Summary	Mono		pos	add	N.A.	obj	N.A.	specificity
	Restatement	Mono		pos	add	N.A.	obj	N.A.	spec.-equiv.
Temporal	Temp.-before	Mono	N-S	pos	add	N.A.	obj	chron	
	Temp.-before	Mono	S-N	pos	add	N.A.	obj	anti	
	Temp.-after	Mono	N-S	pos	add	N.A.	obj	anti	
	Temp.-after	Mono	S-N	pos	add	N.A.	obj	chron	
	Temp.-same-time	Both		pos	add	N.A.	obj	syn	
	Sequence	Multi		pos	add	N.A.	obj	chron	
	Inverted-seq.	Multi		pos	add	N.A.	obj	anti	
Manner-Means	Means	Mono	N-S	pos	cau	non-b	obj	anti	
	Means	Mono	S-N	pos	cau	bas	obj	chron	goal
Topic-Comment	Problem-sol.-n	Mono	N-S	pos	cau	non-b	obj/sub	anti/N.A.	goal
	Problem-sol.-n	Mono	S-N	pos	cau	bas	obj/sub	chron/N.A.	goal
	Problem-sol.-s	Mono	N-S	pos	cau	bas	obj/sub	chron/N.A.	goal
	Problem-sol.-s	Mono	S-N	pos	cau	non-b	obj/sub	anti/N.A.	goal
	Problem-sol.	Multi		pos	cau	bas/non-b	obj/sub	achron/anti/N.A.	goal

Table 9: Mapping of RST relations to UniDim dimensions, taken from Sanders et al. (2018)

Table 9 is the mapping table of relation labels of RST-DT to UniDim dimensions. ***Nuc.*** means the nuclearity of a relation. ***N-S*** means whether the nuclearity is Nucleus-Satellite (N-S) or Satellite-Nucleus (S-N) or Nucleus-Nucleus (N-N). ***Pol.***, ***Basic Op.***, ***Impl. order***, ***Basic Op.***, ***SoC***, ***Temp.***, and ***Add. features*** denote polarity, basic operation, source of coherence, temporality and additional features, respectively.

B Relation Labels of PDTB 3.0 to UniDim Dimension Mapping Table

Table 10 shows the mapping of relation labels of PDTB 3.0 to UniDim dimensions.

Class_type	End label	A1-A2	Pol.	Basic Op.	Impl. order	SoC	Temp.	Add. features
Temporal								
Synchronous			pos	add	N.A.	obj	sync	
Asynchronous	Precedence	A1-A2	pos	add	N.A.	obj	chron	
	Precedence	A2-A1	pos	add	N.A.	obj	anti	
	Succession	A1-A2	pos	add	N.A.	obj	anti	
	Succession	A2-A1	pos	add	N.A.	obj	chron	
Contingency								
Cause	Reason	A1-A2	pos	cau	non-b	obj	anti	
	Reason	A2-A1	pos	cau	bas	obj	chron	
	Result	A1-A2	pos	cau	bas	obj	chron	goal
	Result	A1-A2	pos	cau	bas	obj	chron	goal
	NegResult		neg	cau	bas	obj	chron	
Cause+Belief	Reason+Belief	A1-A2	pos	cau	non-b	sub	NS	
	Reason+Belief	A2-A1	pos	cau	bas	sub	NS	
	Result+Belief	A1-A2	pos	cau	bas	sub	NS	
	Result+Belief	A2-A1	pos	cau	non-b	sub	NS	
Cause +SpeechAct	Reason+SpeechAct	A1-A2	pos	cau	non-b	sub	NS	
	Reason+SpeechAct	A2-A1	pos	cau	bas	sub	NS	
	Result+SpeechAct	A1-A2	pos	cau	bas	sub	NS	
	Result+SpeechAct	A2-A1	pos	cau	non-b	sub	NS	
Purpose	arg1-as-goal	A1-A2	pos	cau	non-b	obj/sub	NS	goal
	arg1-as-goal	A2-A1	pos	cau	bas	obj/sub	NS	goal
	arg2-as-goal	A1-A2	pos	cau	bas	sub	NS	goal
Condition	arg1-as-cond	A1-A2	pos	cau	bas	obj/sub	NS	conditional
	arg1-as-cond	A2-A1	pos	cau	non-b	obj/sub	NS	conditional
	arg2-as-cond	A1-A2	pos	cau	non-b	obj/sub	NS	conditional
	arg2-as-cond	A2-A1	pos	cau	bas	obj/sub	NS	conditional
Condition +SpeechAct			pos	cau	bas	sub	NS	conditional
Negative -Condition	arg1-as-negcond	A1-A2	neg	cau	bas	sub	NS	conditional
	arg1-as-negcond	A2-A1	neg	cau	non-b	sub	NS	conditional
	arg2-as-negcond	A1-A2	neg	cau	non-b	sub	NS	conditional
	arg2-as-negcond	A2-A1	neg	cau	bas	sub	NS	conditional
Negative- Condition+ SpeechAct			neg	cau	bas	sub	NS	conditional
Comparison								
Concession	arg1-as-denier	A1-A2	neg	cau	non-b	obj/sub	NS	
	arg1-as-denier	A2-A1	neg	cau	bas	obj/sub	NS	
	arg2-as-denier	A1-A2	neg	cau	bas	obj/sub	NS	
	arg2-as-denier	A2-A1	neg	cau	non-b	obj/sub	NS	
Concession +SpeechAct			neg	cau	bas	sub	NS	
Contrast			neg	add	NA	obj	NS	
Similarity			pos	add	NA	obj	NS	
Expansion								
Conjunction			pos	add	NA	obj/sub	NS	
Disjunction			neg	add	NA	obj/sub	NS	alternative
Equivalence			pos	add	NA	obj/sub	NS	
Exception	arg1-as-excpt		neg	add	NA	obj/sub	NS	
	arg2-as-excpt		neg	add	NA	obj/sub	NS	
Instantiation	arg1-as-instance		pos	add	NA	obj/sub	NS	specificity
	arg2-as-instance		pos	add	NA	obj/sub	NS	specificity
Level-of-detail	arg1-as-detail		pos	add	NA	obj/sub	NS	specificity
	arg2-as-detail		pos	add	NA	obj/sub	NS	specificity
Manner	arg1-as-manner	A1-A2	pos	add	NA	obj/sub	NS	specificity
	arg2-as-manner		pos	add	NA	obj/sub	NS	specificity
Substitution	arg1-as-subst	A1-A2	neg	cau	bas	obj/sub	NS	
	arg1-as-subst	A2-A1	neg	cau	non-b	obj/sub	NS	
	arg2-as-subst	A1-A2	neg	cau	non-b	obj/sub	NS	
	arg2-as-subst	A2-A1	neg	cau	bas	obj/sub	NS	

Table 10: Mapping of relations labels of PDTB 3.0 to UniDim dimensions.

Table 10 is the mapping table of relation labels of PDTB 3.0 to UniDim dimensions. A1-A2 means Argument 1 precedes Argument 2 and A2-A1 means Argument 2 precedes Argument 1 in the original text. The abbreviations are interpreted in the same way as in Table 9.

C Distribution of UniDim dimensions in RST-DT and PDTB 3.0

Figure 1 shows distribution of the polarity, basic operation, implication order, source of coherence, temporality and additional dimensions used in this paper.

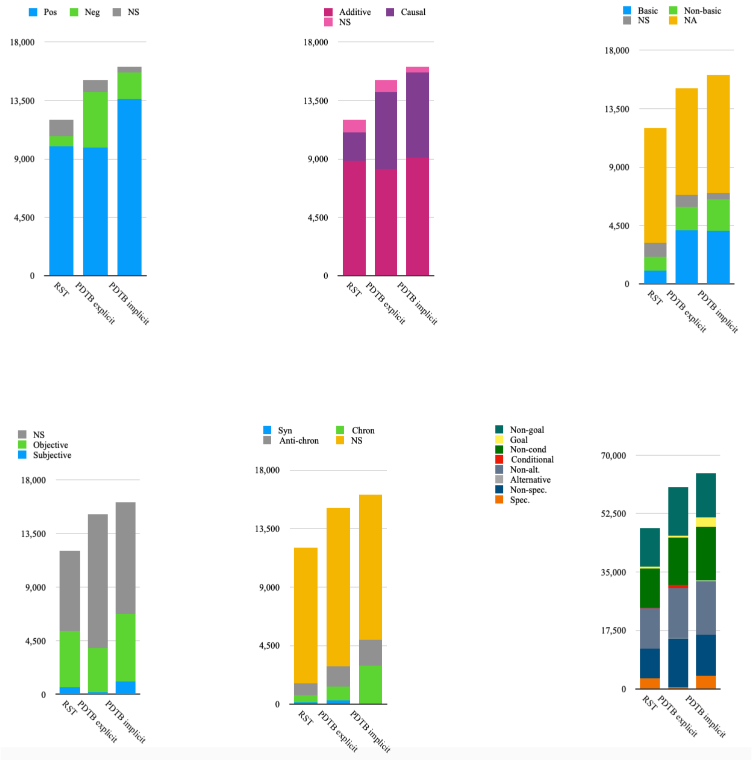


Figure 1: Distribution of the polarity, basic operation, and implication order dimensions (upper row, from left to right, respectively), and source of coherence, temporality and additional dimensions (lower row, from left to right, respectively) in the training sets of RST-DT and PDTB 3.0. We divide PDTB 3.0 based on explicit and implicit relation types.

D Hyper-parameters

For discourse relation classification described in section 3.1, the model is configured with a dropout rate of 0.2. The size of the output of the first MLP is set to 256 and the size of the second MLP output is 128. The model is trained with the AdamW optimizer (Loshchilov and Hutter, 2019), with a learning rate of $5e-5$. The batch size is set to 4 and the maximum norm of gradient clipping is set to 1. We use get_linear_schedule_with_warmup from the Transformers library as the learning rate scheduler. The maximum training epoch number is set to 10. The same setting is used in training the model for UniDim dimension prediction, the only exception being the learning rate, which is set to $1e-5$ to obtain good performance for this task.

For the cross-framework discourse relation classification task, the learning rate for transfer learning is $1e-5$ and as only parameters of the classifier layer are learnable, the maximum training epoch number is set to 50. The other hyper-parameters are the same as above.

We choose the best-performing model based on the performance at the validation set. The PyTorch library (Paszke et al., 2019) is used for implementation. The models are trained on an RTX2060 Super GPU.

The model for PDTB relation classification has 109,753,388 parameters and the training process took 6:25:23 (h:mm:ss) GPU hours for PDTB total relation classification, 2:56:58 GPU hours for PDTB explicit relation classification and 3:13:13 GPU hours for PDTB implicit relation classification. The model for RST relation classification has 109,494,544 parameters and the training process took 2:28:44 GPU hours. The number of parameters in the model for transfer learning is 2,064 and the training process took 4:38:43 GPU hours.

E Distribution of Relations in Training Data

Figures 2, 3, 4 and 5 shows the distribution of relations in the training sets used in the experiments, sorted in descending order.

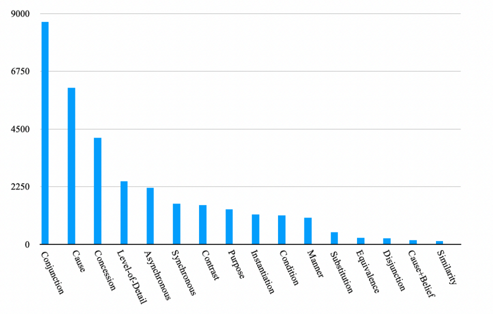


Figure 2: Distribution of PDTB relations in the experiment on PDTB where data of explicit and implicit relations are combined.

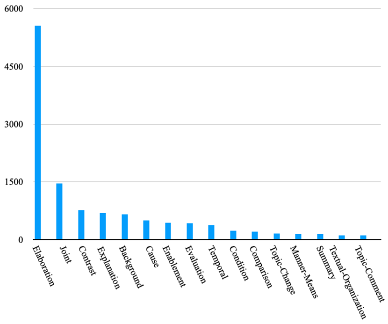


Figure 3: Distribution of RST relations in the training set.

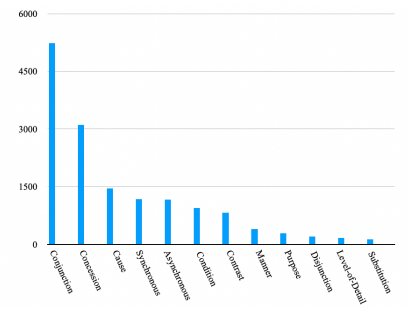


Figure 4: Distribution of PDTB explicit relations in the training set.

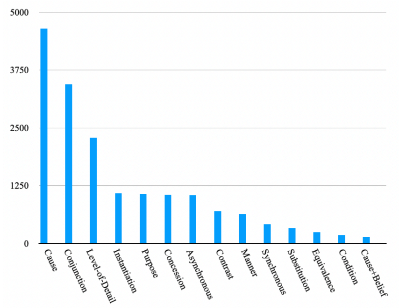


Figure 5: Distribution of PDTB implicit relations in the training set.

F PDTB Total Data Relation Classification

Table 11 shows the classification report on PDTB 3.0 (combining explicit and implicit relations) with BERT embeddings and UniDim dimensions as input features.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	232
Cause	1.00	1.00	1.00	538
Cause+Belief	1.00	1.00	1.00	13
Concession	0.99	0.96	0.98	371
Condition	1.00	1.00	1.00	79
Conjunction	0.97	1.00	0.98	745
Contrast	1.00	1.00	1.00	102
Disjunction	1.00	1.00	1.00	20
Equivalence	0.00	0.00	0.00	25
Instantiation	0.00	0.00	0.00	117
Level-of-detail	0.00	0.00	0.00	202
Manner	0.07	0.96	0.14	26
Purpose	1.00	0.96	0.98	118
Similarity	0.00	0.00	0.00	12
Substitution	0.68	0.91	0.78	35
Synchronous	0.90	1.00	0.95	170
Accuracy	**0.86**			
Macro-F1	**0.66**	**0.74**	**0.67**	2805

Table 11: PDTB relation classification with BERT embeddings and UniDim dimensions as features.

Table 12 shows the classification report on PDTB 3.0 (combining explicit and implicit relations) with BERT embeddings as input.

	Precision	Recall	F1	Support
Asynchronous	0.79	0.65	0.71	232
Cause	0.71	0.62	0.66	538
Cause+Belief	0.00	0.00	0.00	13
Concession	0.78	0.83	0.80	371
Condition	0.92	0.87	0.90	79
Conjunction	0.71	0.85	0.77	745
Contrast	0.48	0.40	0.44	102
Disjunction	0.86	0.90	0.88	20
Equivalence	0.36	0.16	0.22	25
Instantiation	0.70	0.57	0.63	117
Level-of-detail	0.48	0.53	0.50	202
Manner	0.41	0.62	0.49	26
Purpose	0.87	0.84	0.85	118
Similarity	0.78	0.58	0.67	12
Substitution	0.53	0.49	0.51	35
Synchronous	0.74	0.64	0.68	170
Accuracy	**0.71**			
Macro-F1	**0.63**	**0.60**	**0.61**	2805

Table 12: PDTB relation classification with BERT embeddings as features.

G PDTB Explicit Relation Classification

Table 13 shows the classification report on PDTB 3.0 (explicit relations only) with BERT embeddings and UniDim dimensions as input features.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	127
Cause	1.00	1.00	1.00	115
Concession	0.96	1.00	0.98	285
Condition	1.00	1.00	1.00	61
Conjunction	1.00	1.00	1.00	516
Contrast	1.00	1.00	1.00	50
Disjunction	1.00	1.00	1.00	18
Level-of-detail	0.00	0.00	0.00	20
Manner	0.35	1.00	0.52	11
Purpose	1.00	1.00	1.00	29
Substitution	0.00	0.00	0.00	13
Synchronous	1.00	1.00	1.00	126
Accuracy	**0.98**			
Macro-F1	**0.78**	**0.83**	**0.79**	1371

Table 13: Classification report of PDTB explicit relations with BERT embeddings and UniDim dimensions as features.

Table 14 shows the classification report on PDTB 3.0 (explicit relations only) with BERT embeddings as input features.

	Precision	Recall	F1	Support
Asynchronous	0.97	0.87	0.92	127
Cause	0.82	0.89	0.85	115
Concession	0.89	0.95	0.92	285
Condition	0.93	0.92	0.93	61
Conjunction	0.97	0.96	0.96	516
Contrast	0.52	0.48	0.50	50
Disjunction	0.90	1.00	0.95	18
Level-of-detail	0.71	0.75	0.73	20
Manner	0.42	0.91	0.57	11
Purpose	0.62	0.45	0.52	29
Substitution	1.00	0.92	0.96	13
Synchronous	0.81	0.71	0.76	126
Accuracy	**0.89**			
Macro-F1	**0.80**	**0.82**	**0.80**	1371

Table 14: Classification report of PDTB explicit relations with BERT embeddings as features.

H PDTB Explicit Relation Classification Ablation Studies

Table 15 shows the classification report on PDTB 3.0 (explicit relations only) with BERT embeddings and UniDim dimensions as input features, the polarity dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	127
Cause	1.00	1.00	1.00	115
Concession	0.96	1.00	0.98	285
Condition	1.00	1.00	1.00	61
Conjunction	1.00	1.00	1.00	516
Contrast	0.62	1.00	0.76	50
Disjunction	1.00	1.00	1.00	18
Level-of-detail	0.00	0.00	0.00	20
Manner	0.35	1.00	0.52	11
Purpose	1.00	1.00	1.00	29
Substitution	0.00	0.00	0.00	13
Synchronous	1.00	0.75	0.86	126
Accuracy	**0.95**			
Macro-F1	**0.74**	**0.81**	**0.76**	1371

Table 15: Classification report of PDTB explicit relations, with the polarity dimension removed.

Table 16 shows the classification report on PDTB 3.0 (explicit relations only) with BERT embeddings and UniDim dimensions as input features, the basic operation dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	127
Cause	1.00	1.00	1.00	115
Concession	0.96	1.00	0.98	285
Condition	1.00	1.00	1.00	61
Conjunction	1.00	1.00	1.00	516
Contrast	1.00	1.00	1.00	50
Disjunction	1.00	1.00	1.00	18
Level-of-detail	0.00	0.00	0.00	20
Manner	0.35	1.00	0.52	11
Purpose	1.00	1.00	1.00	29
Substitution	0.00	0.00	0.00	13
Synchronous	1.00	1.00	1.00	126
Accuracy	**0.98**			
Macro-F1	**0.78**	**0.83**	**0.79**	1371

Table 16: Classification report of PDTB explicit relations, with the basic operation dimension removed.

Table 17 shows the classification report on PDTB 3.0 (explicit relations only) with BERT embeddings and UniDim dimensions as input features, the source of coherence dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	127
Cause	1.00	1.00	1.00	115
Concession	0.96	1.00	0.98	285
Condition	1.00	1.00	1.00	61
Conjunction	0.94	1.00	0.97	516
Contrast	0.74	1.00	0.85	50
Disjunction	0.00	0.00	0.00	18
Level-of-detail	0.00	0.00	0.00	20
Manner	0.35	1.00	0.52	11
Purpose	1.00	1.00	1.00	29
Substitution	0.00	0.00	0.00	13
Synchronous	1.00	0.75	0.86	126
Accuracy	**0.94**			
Macro-F1	**0.67**	**0.73**	**0.68**	1371

Table 17: Classification report of PDTB explicit relations, with the source of coherence dimension removed.

Table 18 shows the classification report on PDTB 3.0 (explicit relations only) with BERT embeddings and UniDim dimensions as input features, the implication order dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	127
Cause	1.00	1.00	1.00	115
Concession	0.96	1.00	0.98	285
Condition	1.00	1.00	1.00	61
Conjunction	1.00	1.00	1.00	516
Contrast	1.00	1.00	1.00	50
Disjunction	1.00	1.00	1.00	18
Level-of-detail	0.00	0.00	0.00	20
Manner	0.35	1.00	0.52	11
Purpose	1.00	1.00	1.00	29
Substitution	0.00	0.00	0.00	13
Synchronous	1.00	1.00	1.00	126
Accuracy	**0.98**			
Macro-F1	**0.78**	**0.83**	**0.79**	1371

Table 18: Classification report of PDTB explicit relations, with the implication order dimension removed.

Table 19 shows the classification report on PDTB 3.0 (explicit relations only) with BERT embeddings and UniDim dimensions as input features, the temporality dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	0.80	1.00	0.89	127
Cause	1.00	1.00	1.00	115
Concession	0.96	1.00	0.98	285
Condition	1.00	1.00	1.00	61
Conjunction	1.00	1.00	1.00	516
Contrast	1.00	1.00	1.00	50
Disjunction	1.00	1.00	1.00	18
Level-of-detail	0.00	0.00	0.00	20
Manner	0.35	1.00	0.52	11
Purpose	1.00	1.00	1.00	29
Substitution	0.00	0.00	0.00	13
Synchronous	1.00	0.75	0.86	126
Accuracy	**0.95**			
Macro-F1	**0.76**	**0.81**	**0.77**	1371

Table 19: Classification report of PDTB explicit relations, with the temporality dimension removed.

Table 20 shows the classification report on PDTB 3.0 (explicit relations only) with BERT embeddings and UniDim dimensions as input features, the additional dimensions being removed.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	127
Cause	1.00	1.00	1.00	115
Concession	0.96	1.00	0.98	285
Condition	0.88	1.00	0.94	61
Conjunction	0.94	1.00	0.97	516
Contrast	1.00	1.00	1.00	50
Disjunction	1.00	1.00	1.00	18
Level-of-detail	0.00	0.00	0.00	20
Manner	0.00	0.00	0.00	11
Purpose	1.00	0.72	0.84	29
Substitution	0.00	0.00	0.00	13
Synchronous	1.00	1.00	1.00	126
Accuracy	**0.96**			
Macro-F1	**0.73**	**0.73**	**0.73**	1371

Table 20: Classification report of PDTB explicit relations, with the additional dimensions removed.

I PDTB Implicit Relation Classification

Table 21 shows the classification report on PDTB 3.0 (implicit relations only) with BERT embeddings and UniDim dimensions as input features.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	95
Cause	1.00	1.00	1.00	366
Cause+Belief	1.00	0.42	0.59	12
Concession	1.00	0.92	0.96	84
Condition	1.00	1.00	1.00	12
Conjunction	0.90	1.00	0.95	221
Contrast	0.98	1.00	0.99	50
Equivalence	0.00	0.00	0.00	24
Instantiation	0.00	0.00	0.00	107
Level-of-detail	0.60	1.00	0.75	180
Manner	0.00	0.00	0.00	15
Purpose	0.92	0.94	0.93	88
Substitution	0.75	1.00	0.86	21
Synchronous	0.87	0.97	0.92	40
Accuracy	**0.87**			
Macro-F1	**0.72**	**0.73**	**0.71**	1315

Table 21: Classification report of implicit PDTB relations with BERT embeddings and UniDim dimensions as features.

Table 22 shows the classification report on PDTB 3.0 (implicit relations only) with only BERT embeddings as input features.

	Precision	Recall	F1	Support
Asynchronous	0.62	0.61	0.62	95
Cause	0.60	0.63	0.61	366
Cause+Belief	0.00	0.00	0.00	12
Concession	0.44	0.40	0.42	84
Condition	0.71	0.42	0.53	12
Conjunction	0.49	0.61	0.54	221
Contrast	0.45	0.42	0.43	50
Equivalence	0.12	0.04	0.06	24
Instantiation	0.77	0.54	0.64	107
Level-of-detail	0.45	0.48	0.46	180
Manner	0.38	0.60	0.46	15
Purpose	0.92	0.98	0.95	88
Substitution	0.43	0.48	0.45	21
Synchronous	0.27	0.10	0.15	40
Accuracy	**0.56**			
Macro-F1	**0.48**	**0.45**	**0.45**	1315

Table 22: Classification report of PDTB implicit relations with only BERT embeddings as features.

J PDTB Implicit Relation Classification Ablation Studies

Table 23 shows the classification report on PDTB 3.0 (implicit relations only) with BERT embeddings and UniDim dimensions as input features, the polarity dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	95
Cause	1.00	1.00	1.00	366
Cause+Belief	1.00	0.42	0.59	12
Concession	0.96	0.92	0.94	84
Condition	1.00	0.75	0.86	12
Conjunction	0.90	1.00	0.95	221
Contrast	0.98	1.00	0.99	50
Equivalence	0.00	0.00	0.00	24
Instantiation	0.00	0.00	0.00	107
Level-of-detail	0.60	1.00	0.75	180
Manner	0.00	0.00	0.00	15
Purpose	0.92	0.94	0.93	88
Substitution	0.75	1.00	0.86	21
Synchronous	0.87	0.97	0.92	40
Accuracy	**0.87**			
Macro-F1	**0.71**	**0.71**	**0.70**	1315

Table 23: Classification report of PDTB implicit relations, with the polarity dimension removed.

Table 24 shows the classification report on PDTB 3.0 (implicit relations only) with BERT embeddings and UniDim dimensions as input features, the basic operation dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	95
Cause	1.00	1.00	1.00	366
Cause+Belief	1.00	0.42	0.59	12
Concession	1.00	0.92	0.96	84
Condition	1.00	1.00	1.00	12
Conjunction	0.90	1.00	0.95	221
Contrast	1.00	1.00	1.00	50
Equivalence	0.00	0.00	0.00	24
Instantiation	0.00	0.00	0.00	107
Level-of-detail	0.60	1.00	0.75	180
Manner	0.00	0.00	0.00	15
Purpose	0.92	0.94	0.93	88
Substitution	0.75	1.00	0.86	21
Synchronous	0.87	0.97	0.92	40
Accuracy	**0.87**			
Macro-F1	**0.72**	**0.73**	**0.71**	1315

Table 24: Classification report of PDTB implicit relations, with the basic operation dimension removed.

Table 25 shows the classification report on PDTB 3.0 (implicit relations only) with BERT embeddings and UniDim dimensions as input features, the source of coherence dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	95
Cause	1.00	1.00	1.00	366
Cause+Belief	1.00	0.42	0.59	12
Concession	1.00	0.92	0.96	84
Condition	1.00	1.00	1.00	12
Conjunction	0.90	1.00	0.95	221
Contrast	1.00	1.00	1.00	50
Equivalence	0.00	0.00	0.00	24
Instantiation	0.00	0.00	0.00	107
Level-of-detail	0.60	1.00	0.75	180
Manner	0.00	0.00	0.00	15
Purpose	0.92	0.94	0.93	88
Substitution	0.75	1.00	0.86	21
Synchronous	0.87	0.97	0.92	40
Accuracy	**0.87**			
Macro-F1	**0.72**	**0.73**	**0.71**	1315

Table 25: Classification report of PDTB implicit relations, with the source of coherence dimension removed. The result is the same as Table 24, where the basic operation dimension is removed.

Table 26 shows the classification report on PDTB 3.0 (implicit relations only) with BERT embeddings and UniDim dimensions as input features, the implication order dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	1.00	1.00	1.00	95
Cause	1.00	1.00	1.00	366
Cause+Belief	0.00	0.00	0.00	12
Concession	0.80	1.00	0.89	84
Condition	1.00	1.00	1.00	12
Conjunction	0.90	1.00	0.95	221
Contrast	0.98	1.00	0.99	50
Equivalence	0.00	0.00	0.00	24
Instantiation	0.00	0.00	0.00	107
Level-of-detail	0.60	1.00	0.75	180
Manner	0.00	0.00	0.00	15
Purpose	0.87	0.94	0.91	88
Substitution	0.00	0.00	0.00	21
Synchronous	0.87	0.97	0.92	40
Accuracy	**0.86**			
Macro-F1	**0.57**	**0.64**	**0.60**	1315

Table 26: Classification report of PDTB implicit relations, with the implication order dimension removed.

Table 27 shows the classification report on PDTB 3.0 (implicit relations only) with BERT embeddings and UniDim dimensions as input features, the temporality dimension being removed.

	Precision	Recall	F1	Support
Asynchronous	0.99	1.00	0.99	95
Cause	1.00	1.00	1.00	366
Cause+Belief	1.00	0.42	0.59	12
Concession	1.00	0.92	0.96	84
Condition	1.00	1.00	1.00	12
Conjunction	0.90	1.00	0.95	221
Contrast	1.00	1.00	1.00	50
Equivalence	0.00	0.00	0.00	24
Instantiation	0.00	0.00	0.00	107
Level-of-detail	0.60	1.00	0.75	180
Manner	0.00	0.00	0.00	15
Purpose	0.92	0.94	0.93	88
Substitution	0.75	1.00	0.86	21
Synchronous	0.87	0.97	0.92	40
Accuracy	**0.87**			
Macro-F1	**0.72**	**0.73**	**0.71**	1315

Table 27: Classification report of PDTB implicit relations, with the temporality dimension removed.

Table 28 shows the classification report on PDTB 3.0 (implicit relations only) with BERT embeddings and UniDim dimensions as input features, the additional dimensions being removed.

	Precision	Recall	F1	Support
Asynchronous	0.99	1.00	0.99	95
Cause	1.00	1.00	1.00	366
Cause+Belief	1.00	0.42	0.59	12
Concession	0.96	0.92	0.94	84
Condition	1.00	0.75	0.86	12
Conjunction	0.40	1.00	0.58	221
Contrast	1.00	1.00	1.00	50
Equivalence	0.00	0.00	0.00	24
Instantiation	0.00	0.00	0.00	107
Level-of-detail	0.00	0.00	0.00	180
Manner	0.00	0.00	0.00	15
Purpose	0.92	0.94	0.93	88
Substitution	0.75	1.00	0.86	21
Synchronous	0.87	0.97	0.92	40
Accuracy	**0.73**			
Macro-F1	**0.64**	**0.64**	**0.62**	1315

Table 28: Classification report of PDTB implicit relations, with the additional dimensions removed.

K RST Relation Classification

Table 29 shows RST relation classification report with BERT embeddings and UniDim dimensions as input features.

Table 30 shows RST relation classification report with BERT embeddings as input features.

	Precision	Recall	F1	Support
Background	1.00	1.00	1.00	111
Cause	0.92	0.70	0.79	82
Comparison	0.00	0.00	0.00	29
Condition	1.00	1.00	1.00	48
Contrast	0.99	1.00	0.99	146
Elaboration	0.75	1.00	0.86	796
Enablement	0.92	1.00	0.96	46
Evaluation	0.99	1.00	0.99	80
Explanation	0.72	0.97	0.83	110
Joint	1.00	0.03	0.06	212
Manner-Means	0.00	0.00	0.00	27
Summary	0.00	0.00	0.00	32
Temporal	1.00	1.00	1.00	73
Textual-Organization	0.00	0.00	0.00	9
Topic-Change	0.28	1.00	0.44	13
Topic-Comment	0.71	0.21	0.32	24
Accuracy	**0.81**			
Macro-F1	**0.64**	**0.62**	**0.58**	1838

Table 29: RST relation classification report with BERT embeddings and UniDim dimensions as features.

	Precision	Recall	F1	Support
Background	0.47	0.35	0.40	111
Cause	0.50	0.17	0.25	82
Comparison	0.61	0.38	0.47	29
Condition	0.79	0.71	0.75	48
Contrast	0.75	0.68	0.72	146
Elaboration	0.65	0.88	0.75	796
Enablement	0.61	0.85	0.71	46
Evaluation	0.29	0.14	0.19	80
Explanation	0.46	0.27	0.34	110
Joint	0.67	0.62	0.64	212
Manner-Means	0.68	0.48	0.57	27
Summary	0.88	0.47	0.61	32
Temporal	0.74	0.27	0.40	73
Textual-Organization	0.44	0.44	0.44	9
Topic-Change	0.28	0.38	0.32	13
Topic-Comment	0.00	0.00	0.00	24
Accuracy	**0.63**			
Macro-F1	**0.55**	**0.44**	**0.47**	1838

Table 30: RST relation classification report using pre-trained BERT model.

Table 31 shows RST relation classification report using transfer learning from the PDTB relation classification model (combining PDTB explicit and implicit relation data during training) with BERT embeddings and UnDim dimensions as input features.

	Precision	Recall	F1	Support
Background	1.00	1.00	1.00	111
Cause	0.90	0.70	0.79	82
Comparison	0.00	0.00	0.00	29
Condition	1.00	0.98	0.99	48
Contrast	0.99	1.00	0.99	146
Elaboration	0.75	1.00	0.86	796
Enablement	0.92	1.00	0.96	46
Evaluation	1.00	1.00	1.00	80
Explanation	0.72	0.97	0.83	110
Joint	0.00	0.00	0.00	212
Manner-Means	0.00	0.00	0.00	27
Summary	0.00	0.00	0.00	32
Temporal	1.00	1.00	1.00	73
Textual-Organization	0.00	0.00	0.00	9
Topic-Change	0.28	1.00	0.44	13
Topic-Comment	0.71	0.21	0.32	24
Accuracy	**0.81**			
Macro-F1	**0.58**	**0.62**	**0.57**	1838

Table 31: Transfer learning for RST relation classification with the PDTB relation classification model with BERT embeddings and UniDim dimensions as input features.

Table 32 shows RST relation classification report using transfer learning from the pre-trained BERT model fine-tuned on PDTB relation classification task (combining PDTB explicit and implicit relation data).

	Precision	Recall	F1	Support
Background	0.51	0.27	0.35	111
Cause	0.17	0.07	0.10	82
Comparison	0.42	0.38	0.40	29
Condition	0.80	0.67	0.73	48
Contrast	0.75	0.73	0.74	146
Elaboration	0.60	0.82	0.69	796
Enablement	0.48	0.78	0.60	46
Evaluation	0.00	0.00	0.00	80
Explanation	0.40	0.15	0.22	110
Joint	0.57	0.66	0.61	212
Manner-Means	0.43	0.33	0.38	27
Summary	0.00	0.00	0.00	32
Temporal	0.53	0.36	0.43	73
Textual-Organization	0.00	0.00	0.00	9
Topic-Change	0.00	0.00	0.00	13
Topic-Comment	0.00	0.00	0.00	24
Accuracy	**0.58**			
Macro-F1	**0.35**	**0.33**	**0.33**	1838

Table 32: Transfer learning for RST relation classification using BERT embeddings as input.

L RST Relation Classification Ablation Studies

Table 33 shows the classification report on RST-DT with BERT embeddings and UniDim dimensions as input features, the polarity dimension being removed.

	Precision	Recall	F1	Support
Background	1.00	1.00	1.00	111
Cause	0.90	0.70	0.79	82
Comparison	0.00	0.00	0.00	29
Condition	1.00	0.94	0.97	48
Contrast	0.61	0.56	0.58	146
Elaboration	0.68	1.00	0.81	796
Enablement	0.92	1.00	0.96	46
Evaluation	1.00	0.57	0.73	80
Explanation	0.71	0.97	0.82	110
Joint	0.00	0.00	0.00	212
Manner-Means	0.00	0.00	0.00	27
Summary	0.00	0.00	0.00	32
Temporal	1.00	1.00	1.00	73
Textual-organization	0.00	0.00	0.00	9
Topic-Change	0.00	0.00	0.00	13
Topic-Comment	0.00	0.00	0.00	24
Accuracy	**0.74**			
Macro-F1	**0.49**	**0.48**	**0.48**	1838

Table 33: Classification report for RST, with the polarity dimension removed.

Table 34 shows the classification report on RST-DT with BERT embeddings and UniDim dimensions as input features, the basic operation dimension being removed.

	Precision	Recall	F1	Support
Background	0.95	1.00	0.97	111
Cause	0.90	0.70	0.79	82
Comparison	0.00	0.00	0.00	29
Condition	1.00	0.98	0.99	48
Contrast	0.99	1.00	0.99	146
Elaboration	0.73	1.00	0.84	796
Enablement	0.92	1.00	0.96	46
Evaluation	0.87	0.57	0.69	80
Explanation	0.72	0.97	0.83	110
Joint	0.00	0.00	0.00	212
Manner-Means	0.00	0.00	0.00	27
Summary	0.00	0.00	0.00	32
Temporal	1.00	1.00	1.00	73
Textual-Organization	0.00	0.00	0.00	9
Topic-Change	0.28	1.00	0.44	13
Topic-Comment	0.00	0.00	0.00	24
Accuracy	**0.78**			
Macro-F1	**0.52**	**0.58**	**0.53**	1838

Table 34: Classification report for RST, with the basic operation dimension removed.

Table 35 shows the classification report on RST-DT with BERT embeddings and UniDim dimensions as input features, the source of coherence dimension being removed.

	Precision	Recall	F1	Support
Background	0.95	1.00	0.97	111
Cause	0.84	0.70	0.76	82
Comparison	0.00	0.00	0.00	29
Condition	1.00	0.98	0.99	48
Contrast	0.99	1.00	0.99	146
Elaboration	0.73	1.00	0.84	796
Enablement	0.92	1.00	0.96	46
Evaluation	0.96	0.57	0.72	80
Explanation	0.72	0.97	0.83	110
Joint	0.00	0.00	0.00	212
Manner-Means	0.00	0.00	0.00	27
Summary	0.00	0.00	0.00	32
Temporal	1.00	1.00	1.00	73
Textual-Organization	0.00	0.00	0.00	9
Topic-Change	0.28	1.00	0.44	13
Topic-Comment	0.00	0.00	0.00	24
Accuracy		**0.78**		
Macro-F1	**0.52**	**0.58**	**0.53**	1838

Table 35: Classification report for RST, with the source of coherence dimension removed.

Table 36 shows the classification report on RST-DT with BERT embeddings and UniDim dimensions as input features, the implication order dimension being removed.

	Precision	Recall	F1	Support
Background	1.00	1.00	1.00	111
Cause	0.90	0.70	0.79	82
Comparison	0.00	0.00	0.00	29
Condition	1.00	0.98	0.99	48
Contrast	0.99	1.00	0.99	146
Elaboration	0.75	1.00	0.86	796
Enablement	0.84	1.00	0.91	46
Evaluation	0.99	1.00	0.99	80
Explanation	0.72	0.97	0.83	110
Joint	0.75	0.03	0.05	212
Manner-Means	0.00	0.00	0.00	27
Summary	0.00	0.00	0.00	32
Temporal	1.00	1.00	1.00	73
Textual-Organization	0.00	0.00	0.00	9
Topic-Change	0.28	1.00	0.44	13
Topic-Comment	0.00	0.00	0.00	24
Accuracy		**0.81**		
Macro-F1	**0.58**	**0.60**	**0.55**	1838

Table 36: Classification report for RST, with the implication order dimension removed.

Table 37 shows the classification report on RST-DT with BERT embeddings and UniDim dimensions as input features, the temporality dimension being removed.

	Precision	Recall	F1	Support
Background	1.00	1.00	1.00	111
Cause	0.92	0.70	0.79	82
Comparison	0.00	0.00	0.00	29
Condition	1.00	0.88	0.93	48
Contrast	0.99	1.00	0.99	146
Elaboration	0.75	1.00	0.86	796
Enablement	0.84	1.00	0.91	46
Evaluation	0.99	1.00	0.99	80
Explanation	0.69	0.97	0.81	110
Joint	1.00	0.03	0.06	212
Manner-Means	0.00	0.00	0.00	27
Summary	0.00	0.00	0.00	32
Temporal	1.00	1.00	1.00	73
Textual-Organization	0.00	0.00	0.00	9
Topic-Change	0.28	1.00	0.44	13
Topic-Comment	0.00	0.00	0.00	24
Accuracy		**0.80**		
Macro-F1	**0.59**	**0.60**	**0.55**	1838

Table 37: Classification report for RST, with the temporality dimension removed.

Table 38 shows the classification report on RST-DT with BERT embeddings and UniDim dimensions as input features, the additional dimensions being removed.

	Precision	Recall	F1	Support
Background	0.95	1.00	0.97	111
Cause	0.90	0.70	0.79	82
Comparison	0.00	0.00	0.00	29
Condition	1.00	0.81	0.90	48
Contrast	0.99	1.00	0.99	146
Elaboration	0.75	1.00	0.86	796
Enablement	0.84	1.00	0.91	46
Evaluation	0.90	1.00	0.95	80
Explanation	0.71	0.97	0.82	110
Joint	0.00	0.00	0.00	212
Manner-Means	0.00	0.00	0.00	27
Summary	0.00	0.00	0.00	32
Temporal	1.00	1.00	1.00	73
Textual-Organization	0.00	0.00	0.00	9
Topic-Change	0.28	1.00	0.44	13
Topic-Comment	0.00	0.00	0.00	24
Accuracy		**0.80**		
Macro-F1	**0.52**	**0.59**	**0.54**	1838

Table 38: Classification report for RST, with the additional dimensions removed.

M Cross-framework Discourse Relation Classification

Table 39 shows the classification report of the experiment using total PDTB data, where PDTB relation classification is the source task.

	P	R	$F1$	$P_{b.}$	$R_{b.}$	$F1_{b.}$	$C.$
Asynchronous	1.00	1.00	1.00	0.79	0.65	0.71	232
Cause	1.00	1.00	1.00	0.71	0.62	0.66	538
Cause+Belief	1.00	1.00	1.00	0.00	0.00	0.00	13
Concession	0.99	0.96	0.98	0.78	0.83	0.80	371
Condition	1.00	1.00	1.00	0.92	0.87	0.90	79
Conjunction	0.97	1.00	0.98	0.71	0.85	0.77	745
Contrast	1.00	1.00	1.00	0.48	0.40	0.44	102
Disjunction	1.00	1.00	1.00	0.86	0.90	0.88	20
Equivalence	0.00	0.00	0.00	0.36	0.16	0.22	25
Instantiation	0.00	0.00	0.00	0.70	0.57	0.63	117
Level-of-detail	0.00	0.00	0.00	0.48	0.53	0.50	202
Manner	0.07	0.96	0.14	0.41	0.62	0.49	26
Purpose	1.00	0.96	0.98	0.87	0.84	0.85	118
Similarity	0.00	0.00	0.00	0.78	0.58	0.67	12
Substitution	0.68	0.91	0.78	0.53	0.49	0.51	35
Synchronous	0.90	1.00	0.95	0.74	0.64	0.68	170
Acc.		**0.86**		0.71 (vs. DISRPT 2021: 0.74)			
Macro-F1	**0.66**	**0.74**	**0.67**	0.63	0.60	0.61	2805

Table 39: Results of relation classification on total PDTB data. Blue columns show our results and uncolored columns show results of the baseline model.

Table 40 shows the classification report of the target task, i.e. RST relation classification.

	P	R	$F1$	$P_{b.}$	$R_{b.}$	$F1_{b.}$	$C.$
Background	1.00	1.00	1.00	0.51	0.27	0.35	111
Cause	0.90	0.70	0.79	0.17	0.07	0.10	82
Comparison	0.00	0.00	0.00	0.42	0.38	0.40	29
Condition	1.00	0.98	0.99	0.80	0.67	0.73	48
Contrast	0.99	1.00	0.99	0.75	0.73	0.74	146
Elaboration	0.75	1.00	0.86	0.60	0.82	0.69	796
Enablement	0.92	1.00	0.96	0.48	0.78	0.60	46
Evaluation	1.00	1.00	1.00	0.00	0.00	0.00	80
Explanation	0.72	0.97	0.83	0.40	0.15	0.22	110
Joint	0.00	0.00	0.00	0.57	0.66	0.61	212
Manner-Means	0.00	0.00	0.00	0.43	0.33	0.38	27
Summary	0.00	0.00	0.00	0.00	0.00	0.00	32
Temporal	1.00	1.00	1.00	0.53	0.36	0.43	73
Textual-Organization	0.00	0.00	0.00	0.00	0.00	0.00	9
Topic-Change	0.28	1.00	0.44	0.00	0.00	0.00	13
Topic-Comment	0.71	0.21	0.32	0.00	0.00	0.00	24
Acc.		**0.81**			0.58		
Macro-F1	**0.58**	**0.62**	**0.57**	0.35	0.33	0.33	1838
RST acc		0.81			0.63		
RST Macro-F1	0.64	0.62	0.58	0.55	0.44	0.47	1838

Table 40: Results of the target task. The results of training a model specifically for RST relation classification with our method are shown in blue columns and the uncolored columns show results of the baseline model.

Representation Learning for Hierarchical Classification of Entity Titles

Elena Chistova
FRC CSC RAS, Moscow, Russia
chistova@isa.ru

Abstract

We present a method for effective title encoding for hierarchical classification in a large taxonomy. The method enables taxonomy-aware encoding in pre-trained text encoders, such as fastText and BERT, which are additionally fine-tuned for the hierarchical classification. The embeddings produced using our method perform well when applied to nearest neighbor classification. They allow for controllable and sufficient hierarchical classification based solely on the title.

1 Introduction

Hierarchical classification is the task of organizing data into a hierarchy of categories, where each category is a subset of another category. This structure can be thought of as a tree-like structure, where the root node represents the most general category and the leaf nodes represent the most specific categories. In NLP, hierarchical text classification (HTC) is widely used to organize large collections of documents (e.g. emails, patents, job advertisements, digital libraries) or entities (e.g. product or service titles in e-commerce). This work focuses on the challenge of inferring fine-grained categories from no other information but an entity name, which is a specific challenge for hierarchical classification.

The deep hierarchical classification approaches developed over the past years (Yang et al., 2020; Gao, 2020; Gong et al., 2023) have three major limitations:

- Entity HTC models are often developed for e-commerce and use multiple attributes for the input entity including detailed descriptions, tags, or images. However, there are other situations where just the textual titles are available for classification, like mapping diagnoses and procedures to a clinical coding taxonomy (Li et al., 2019; Chakraborty et al., 2023). Better title representations can also be beneficial when multiple attributes are present.
- Being mostly deep learning classification methods, they are prone to class imbalance and may not be able to handle large skewed hierarchies with a few examples per leaf.
- Limited interpretation capabilities of the deep hierarchical classifiers are another disadvantage that can be critical in some practical applications.

To address these limitations, we propose a simple yet effective approach that encodes the textual title using hierarchy-aware information to map an object's title to the relevant leaf in the taxonomy. We show that our approach improves the classification performance of deep models while making the entity title classification easier to interpret and control[1].

2 Related Work

Hierarchical Entity Title Classification In hierarchical classification, each object is associated with a certain branch (labels path) in the hierarchy tree. There are three fundamental approaches to hierarchical classification: flat classification (object-to-branch), global classification, and local classification (Silla and Freitas, 2011). Global classification predicts classes in the hierarchy using a single model that considers class dependencies, whereas local classification uses multiple separate models for different hierarchy nodes or levels.

Previous approaches to HTC for e-commerce mainly focus on title-plus-description classification, and include flat classifiers (Skinner, 2018; Suzuki et al., 2018), two-level pipelines (Cevahir and Murakami, 2016; Gupta et al., 2016; Das et al., 2017; Goumy and Mejri, 2018), multilabel classifiers (Jia et al., 2018; Yu et al., 2018), and sequence-to-sequence branch generation (Li et al., 2018).

[1] The code is available at https://github.com/tchewik/entity_representation_learning

Shared tasks often feature the systems investigating external ways to improve classification performance, including model ensembling (Yang et al., 2020; Yu et al., 2018; Jia et al., 2018), pseudo labeling (Yang et al., 2020), and collecting additional data (Borst et al., 2020). Some approaches focus on optimizing the classification model itself by considering the hierarchy of classes in the activation (Yang et al., 2020) or loss (Gao, 2020) function. Other methods involve matching an entity title with a leaf title (Chen et al., 2021; Gong et al., 2023). To improve entity title encoding for product classification and overcome the problem of domain shift, Brinkmann and Bizer (2021) suggest additionally pre-training the transformer on product offers from Common Crawls.

In our method, we train a single global deep classifier and utilize it to encode entity titles in a complicated hierarchy for flat categorization. We demonstrate that this approach excels in terms of accuracy on the deepest levels of hierarchy, simplicity, and controllability.

LLM Applications Large language models have limited structured prediction capabilities. There have been recent attempts to solve the HTC task through hierarchy verbalization, however, they still rely on pretrained BERT rather than LLMs and require model architecture modifications: Wang et al. (2022) frame the problem as a hierarchy-aware multi-label MLM task, adopting a Graph Attention Network and a zero-bounded Multi-label Cross-Entropy Loss, while Ji et al. (2023) address HTC as flat classification solvable by verbalizing with a hierarchy-aware decoder constraint. Although promising, these methods are tailored and evaluated for elaborate texts in smaller taxonomies (WOS, DBPedia, RCV1-V2).

While prompting LLMs for this task can be possible for flat entity title classification in a large hierarchy, there are some major limitations:

- A large language model should memorize an entire deep taxonomy with thousands of branches and adhere to its complex structure without deviation. This level of precision is achievable by imposing low-level constraints overriding the NLG capabilities of LLMs. Constraining LLMs in this way erases their main strength in favor of precise taxonomic compliance – an outcome more efficiently reached by fine-tuning text encoders.

- Few-shot learning is successful in many tasks, but it is not suitable for the hierarchical classification in a large taxonomy. Exposing the LLM to examples spanning all the taxonomy branches, or fine-tuning on a large labeled dataset, would be extremely resource- and time-intensive.

- LLM predictions cannot be controlled or interpreted precisely. This lack of transparency makes LLMs unsuitable for settings requiring controllable accuracy and recall.

3 Background

In this work, we compare nearest-neighbors classification, deep hierarchical classification, and our hybrid method as three basic approaches to entity title classification in a large taxonomy.

3.1 k-Nearest-Neighbor Classification

Given representations of entity titles in hierarchically organized data, the embedding of an input entity is assigned to a leaf of the hierarchy based on the leaves of its k nearest neighbors. The distance between text embeddings is typically estimated as a cosine distance, and k nearest neighbor classes are weighted according to the distances.

Advantages: (1) The most interpretable method. (2) With small k is immune to subclass imbalance in a complex hierarchy.

Disadvantages: (1) Domain shift affects pretrained language models substantially, and domain adaptation requires additional resources for data collection and computation. (2) With small k, highly sensitive to outliers. (3) Does not provide any information about the taxonomy.

3.2 Deep Hierarchical Classification

The classifier predicts the most probable classes for each level of the hierarchy and collects the final prediction from a pool of weighted class labels. The classifier can predict multiple labels in a multi-label fashion or have n top outputs for all hierarchy levels.

Advantages: (1) The internal representations of texts in the neural model are influenced by both their own surface forms and their position in the hierarchy. (2) More robust to data noise. (3) Can more or less adjust to specific domains while fine-tuning.

Disadvantages: (1) Is highly affected by class imbalance. (2) Has reduced interpretability. (3) As

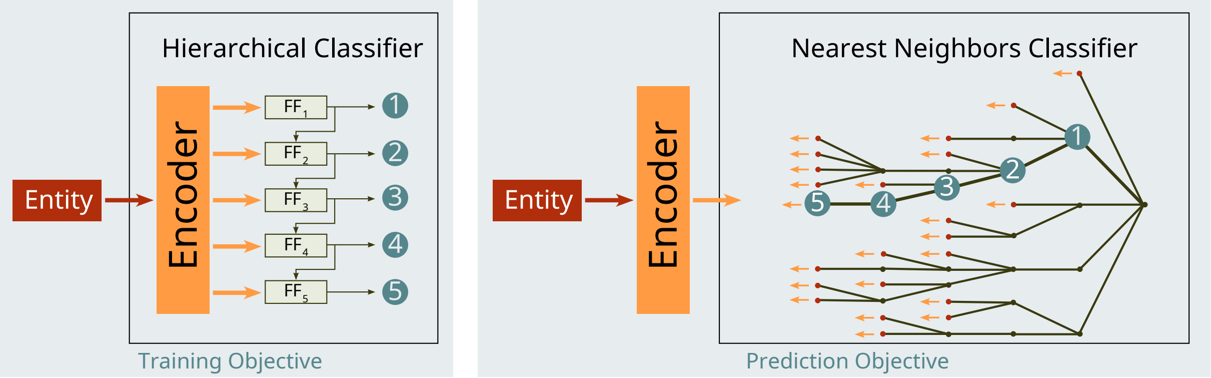


Figure 1: Overview of our framework with DBERT$_{1-5}$ as a deep classifier. During training, the encoder is paired with outputs for hierarchical classification. The classification part of the model is fine-tuned in conjunction with the encoder to generate a sequence of subclass labels (levels 1-5). During inference, we encode the known data and input entity using only the fine-tuned encoder and attempt to find the most similar entities in a complex taxonomy. Finally, we assign the input entity to the hierarchy leaf with the most similar known entities.

a result of the previous point, it is more difficult to control the precision of the model when implementing it in real-world systems. Classifier confidence is not transparent. (3) The model itself can produce contradictory labels (non-existing taxonomy branches), and introducing hierarchical information can require the implementation of additional restrictions.

4 Methods

We compare multiple methods that follow the two fundamental strategies introduced in Section 3. The described HTC methods employing a single model (FT$_{1-5}$, DBERT$_{1-5}$) are additionally probed in the hybrid classification setting.

4.1 k-NN

The most similar titles in a hierarchy are found using cosine distance. The out-of-the-box encoder is not fine-tuned on task-related data. The title is encoded as an average of token representations. We probe two types of representations: **fastText** and **DeBERTa**.

4.2 Trainable Classification

A deep classification model simultaneously predicts multiple labels denoting the nodes in a hierarchy. The final prediction assigns an entity title to a taxonomy branch and is constructed from top-n predicted node labels along with their probabilities.

FT$_{1-5}$: To predict top-n possible nodes for levels 1-5, we use a one-vs-all multilabel classification implemented in the fastText[2] library.

DBERT$_{1-5}$: We use an architecture of a deep hierarchical classifier similar to that of Gao (2020). The output layers for every level are added on top of an encoding language model (DeBERTa). For the title consisting of tokens $w_1, w_2, ..., w_z$, the representations are computed in encoder:

$$e = \text{Encoder}(w_1 w_2 ... w_k) \in \mathbb{R}^{d_{LM}} \qquad (1)$$

The output for each hierarchy level i is predicted with a separate feedforward layer. Input for the output layer $i > 1$ is a concatenation of the text embedding e and an output for the previous level:

$$y_i = \begin{cases} \text{FF}_i(e) & \text{if } i = 1; \\ \text{FF}_i(e \oplus y_{i-1}), & \text{otherwise.} \end{cases} \qquad (2)$$

The probabilities of classes for a hierarchy level i are calculated by passing y_i through the softmax activation function. The class with the highest predicted probability is then predicted as $\hat{y}_i$. The loss function is a weighted sum of the categorical cross-entropy loss and hierarchical loss:

$$\begin{gathered} \text{HLoss}_i = \begin{cases} 0 & \text{if } \hat{y}_i \subset \hat{y}_{i-1}; \\ 1 & \text{otherwise.} \end{cases} \\ \text{Loss} = \alpha \sum_{i=1}^{n} \text{CELoss}_i + \sum_{i=2}^{n} \beta^{i-1} \text{HLoss}_i \end{gathered} \qquad (3)$$

where α and β are the weights controlling the impact of hierarchical loss. The hyperparameter β ($0 < \beta \leq 1$) is used to scale the hierarchical loss. The cross-entropy loss is weighted to handle the class imbalance on each hierarchy level.

[2] https://fasttext.cc/

Part	Deduplicated Length	Unique Branches	Unique Classes of Each Level 1	2	3	4	5
Clothing Shoes and Jewelry	1988301	35710	11	253	953	5263	12371
Home and Kitchen	1203754	1671	13	136	539	695	292
Automotive	831549	2252	14	165	743	849	318
Sports and Outdoors	809999	3414	3	43	351	1102	1281
Electronics	584136	900	16	105	290	305	133
Tools and Home Improvement	488042	1152	13	98	435	427	172
Industrial and Scientific	132168	1796	25	301	970	496	93

Table 1: Statistics of the corpus.

4.3 Our hybrid approach

As a compromise between both of the described methods, we propose a hybrid approach, in which the out-of-the-box text encoder is additionally pretrained on hierarchical classification. The overall framework is illustrated in Figure 1. The title encoder is fine-tuned as a part of a hierarchical classifier, and the nearest-neighbor classifier dealing with flat $(\text{entity}; \text{leaf})$ pairs predicts the leaf with most similar entities.

5 Experimental Setup

5.1 Dataset

Deep models with millions of parameters, such as BERT, have a tendency to overfit to noise and outliers in e-commerce product classification data, as noted by Zhang et al. (2021). They describe two major challenges in e-commerce data: frequently incomplete or misleading item descriptions and confusing or non mutually exclusive labels in a large taxonomy. Supervised learning faces a significant obstacle when classifying images, descriptions, or titles due to confusing and non-mutually exclusive labels in a large taxonomy. To address this issue, we thoroughly clean the data for our experiments.

We only use the titles and hierarchy annotations from the Amazon review dataset[3] (Ni et al., 2019); HTML character references in both titles and categories are decoded into Unicode. We cut subbranches leaving only the nodes containing less than 13 tokens[4] in name and keep only subbranches appearing in the data at least 4 times. We have selected seven major data subsets that have at least 90 classes annotated in the 5th level of the hierarchy. The statistics of the obtained data are described in Table 1. On each hierarchy level, we encode classes independently of the previous levels. As a result, on most subsets, the number of classes decreases after level 4; instead, “missing” class replacement occurs most frequently. This denotes a natural skew in the hierarchy.

5.2 Metrics

We evaluate the hierarchical classification performance with 5-fold stratified cross-validation. This balances the distribution of branches in each fold. Firstly, we calculate macro-averaged F1 for each level of the hierarchy. Since this F1 reflects performance for each level independently, we also evaluate the accuracy for flat branch assignment for each depth.

5.3 Implementation Details

fastText We use a fastText model described in (Grave et al., 2018) that is pretrained on Common Crawl and Wikipedia data. *Hierarchical model* (FT_{1-5})*:* The classifier is fine-tuned using the one-vs-all scheme, with a learning rate of 1, character n-gram range of (3, 10), and for 25 epochs. The top 7 predicted nodes are used to assemble the full branch after classification.

Contextual Embeddings As a pretrained transformer, we employ DeBERTa[5] (He et al., 2021). *Hierarchical model* (DBERT_{1-5})*:* The model is fine-tuned with a learning rate of 2e-5, dropout rate of 0.4, batch size of 128, $\alpha = 1$, $\beta = 0.9$, and the cross-entropy loss for each level (CELoss_i in (3)) is weighted based on the distribution of classes in the subcorpus. The top 8 predicted nodes are used to assemble the final branch.

[3] https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/

[4] We considered the longer nodes noisy because they often included non-taxonomy information, such as notes for customers (e.g. “*Please feel free to contact us if you have any special requests or questions*”) or lengthy keyword-stuffed descriptions (e.g. “*My Daily Styles Stainless Steel Black Faux PU Leather Yellow Gold-Tone Latin Cross Religious Adjustable Wristband Mens Bracelet*”) hardly resembling subclasses.

[5] microsoft/deberta_v3_base

6 Experimental Results

Table 2 compares all the investigated methods for hierarchical classification. The statistics of macro F1 calculated for each level independently are illustrated in Figure 2.

6.1 Baselines

The results for kNN using out-of-the-box pretrained text encoders are denoted as KNN:FT for fastText and KNN:DBERT for DeBERTa. The fastText-based flat kNN classifier provides a strong baseline across all subcorpora. The low performance of the KNN:DBERT can be attributed to a known issue with transformers: the feature extraction performance of the frozen model decreases with increasing difference between pretraining and target tasks (Peters et al., 2019).

6.2 Trainable Classifiers

The fastText- and DeBERTa-based classifiers are denoted as FT$_{1-5}$ and DBERT$_{1-5}$, respectively.

According to the results in Table 2, the fastText-based hierarchical classifier outperforms the kNN baseline only across the smallest subcorpora, and mostly for the higher levels of hierarchy. Moreover, for larger datasets, starting with "Sports and Outdoors" multilabel fastText training becomes increasingly more challenging and consuming. The statistics of hierarchical labels are actually learned by the model, which we'll see by applying kNN to its representations. However, collecting the taxonomy branch from top-n pool of predicted labels using the direct approach is hardly applicable.

DBERT$_{1-5}$ outperforms not only the corresponding weak baseline but also the fasttext-based hybrid classification KNN:FT$_{1-5}$ on many datasets. It is also worth noting that this method handles larger data with larger class sets much better than multilabel fastText.

6.3 k-NN over the Tuned Representations

Applying kNN directly to the inner representations results in an improvement in classification for all levels for both backbones (KNN:FT$_{1-5}$ and KNN:DBERT$_{1-5}$). In addition to a considerable improvement in the accuracy of full branch prediction (A$_{1-5}$ in Table 2) while preserving or improving the intra-level F1 (Figure 2), the purely vector-based approach can also be significantly faster than collecting known branches from a pool of predicted labels for each entity.

	A$_1$	A$_{1-2}$	A$_{1-3}$	A$_{1-4}$	A$_{1-5}$
	Clothing Shoes and Jewelry				
KNN:FT	85.5	81.4	71.0	55.3	44.4
FT$_{1-5}$	84.1	76.8	64.1	45.4	31.1
KNN:FT$_{1-5}$	86.8	83.1	73.3	57.8	45.9
KNN:DBERT	72.7	64.3	51.3	38.9	31.8
DBERT$_{1-5}$	90.8	88.2	80.3	65.7	52.7
KNN:DBERT$_{1-5}$	**90.9**	**88.4**	**80.8**	**67.0**	**54.9**
	Home and Kitchen				
KNN:FT	89.7	78.5	68.5	64.4	63.5
FT$_{1-5}$	91.5	80.0	68.0	62.9	60.6
KNN:FT$_{1-5}$	90.9	81.0	71.5	67.4	66.5
KNN:DBERT	64.5	49.6	42.2	39.8	39.8
DBERT$_{1-5}$	93.3	85.0	76.5	72.7	71.6
KNN:DBERT$_{1-5}$	**93.6**	**85.6**	**77.6**	**74.0**	**73.1**
	Automotive				
KNN:FT	89.4	82.1	76.1	72.7	72.0
FT$_{1-5}$	88.5	80.1	72.9	67.9	66.6
KNN:FT$_{1-5}$	91.8	86.0	80.7	77.4	76.7
KNN:DBERT	76.9	66.9	61.3	58.7	58.3
DBERT$_{1-5}$	92.1	86.3	80.8	77.4	76.6
KNN:DBERT$_{1-5}$	**92.3**	**86.9**	**81.8**	**78.6**	**77.8**
	Sports and Outdoors				
KNN:FT	91.8	81.7	73.0	64.2	59.3
FT$_{1-5}$	90.3	78.0	67.1	56.7	50.3
KNN:FT$_{1-5}$	93.3	85.2	77.5	69.2	64.6
KNN:DBERT	77.0	54.5	46.0	40.8	38.0
DBERT$_{1-5}$	94.4	87.3	80.3	72.6	68.0
KNN:DBERT$_{1-5}$	**94.5**	**87.8**	**81.2**	**74.0**	**69.8**
	Electronics				
KNN:FT	87.0	76.3	68.6	64.0	62.6
FT$_{1-5}$	87.4	74.6	64.8	58.3	56.7
KNN:FT$_{1-5}$	89.4	79.8	72.6	68.5	67.2
KNN:DBERT	63.9	50.3	43.6	40.4	39.6
DBERT$_{1-5}$	89.8	80.1	72.5	68.1	66.9
KNN:DBERT$_{1-5}$	**90.1**	**80.8**	**73.7**	**69.5**	**68.3**
	Tools and Home Improvement				
KNN:FT	88.3	78.9	68.4	64.3	62.9
FT$_{1-5}$	89.9	79.8	69.2	63.7	62.1
KNN:FT$_{1-5}$	91.9	84.3	75.2	70.9	69.6
KNN:DBERT	62.0	51.7	43.9	41.7	40.8
DBERT$_{1-5}$	92.2	84.6	75.6	71.1	69.8
KNN:DBERT$_{1-5}$	**92.4**	**85.2**	**76.5**	**72.2**	**70.9**
	Industrial and Scientific				
KNN:FT	82.0	71.8	63.6	60.6	60.2
FT$_{1-5}$	85.1	74.1	64.7	60.3	59.7
KNN:FT$_{1-5}$	87.9	79.1	71.4	68.2	67.8
KNN:DBERT	56.8	49.0	44.1	42.4	42.3
DBERT$_{1-5}$	88.2	78.9	70.6	67.4	67.0
KNN:DBERT$_{1-5}$	**88.4**	**79.4**	**71.5**	**68.5**	**68.0**

Table 2: Mean accuracy of the branch prediction. The datasets are listed in descending order of size (see Table 1).

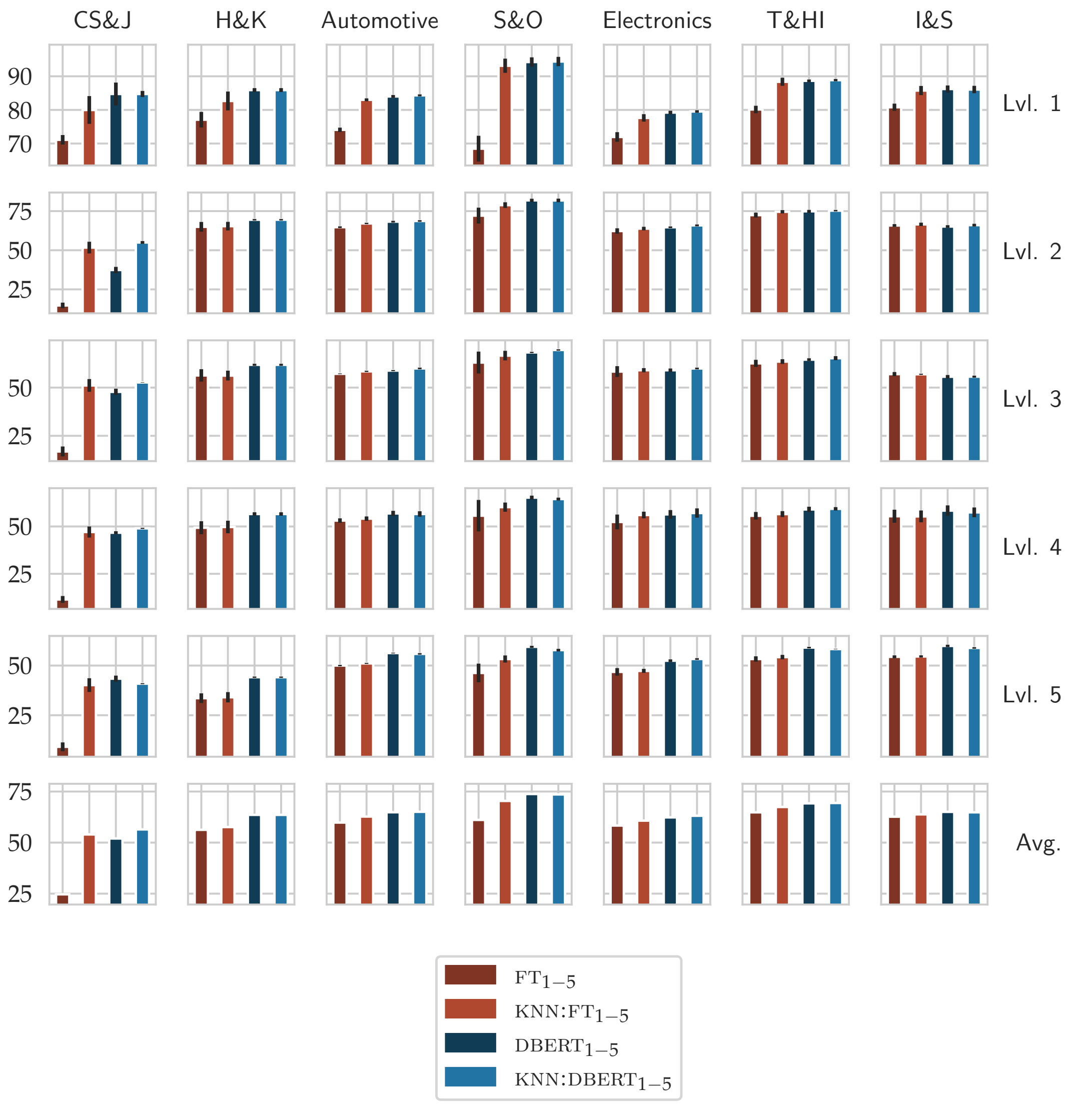


Figure 2: Macro F1 calculated for each level independently; two types of classifiers.

7 Conclusion

We present an approach for entity title hierarchical classification that uses representation learning for training hierarchy-informed embeddings. We apply the obtained embeddings in kNN flat hierarchical classification to demonstrate how these representations can be directly used in a controllable setting. The baselines include pretrained encoders used as the base encoders in the pipeline and hierarchical classifiers built with the same encoders. The hybrid approach outperforms the baselines on each part of the large-taxonomy e-commerce corpus.

Acknowledgments

The research was carried out using the infrastructure of the Shared Research Facilities «High Performance Computing and Big Data» (CKP «Informatics») of FRC CSC RAS (Moscow).

References

Janos Borst, Erik Korner, Kobkaew Opasjumruskit, and Andreas Niekler. 2020. Language model CNN-driven similarity matching and classification for HTML-embedded product data. In *Proceedings of the semantic web challenge on mining the web of HTML-embedded product data co-located with the 19th international semantic web conference*.

Alexander Brinkmann and Christian Bizer. 2021. Improving hierarchical product classification using domain-specific language modelling. In *Proceedings of Workshop on Knowledge Management in e-Commerce @ The Web Conference '21*, volume 44, pages 14–25.

Ali Cevahir and Koji Murakami. 2016. Large-scale multi-class and hierarchical product categorization for an E-commerce giant. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 525–535, Osaka, Japan. The COLING 2016 Organizing Committee.

Sinchani Chakraborty, Harsh Raj, Srishti Gureja, Tanmay Jain, Atif Hassan, and Sayantan Basu. 2023. Evaluating the robustness of biomedical concept normalization. In *Proceedings of The 1st Transfer Learning for Natural Language Processing Workshop*, pages 63–73. PMLR.

Haibin Chen, Qianli Ma, Zhenxi Lin, and Jiangyue Yan. 2021. Hierarchy-aware label semantics matching network for hierarchical text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4370–4379, Online. Association for Computational Linguistics.

Pradipto Das, Yandi Xia, Aaron Levine, Giuseppe Di Fabbrizio, and Ankur Datta. 2017. Web-scale language-independent cataloging of noisy product listings for E-commerce. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 969–979, Valencia, Spain. Association for Computational Linguistics.

Dehong Gao. 2020. Deep hierarchical classification for category prediction in E-commerce system. In *Proceedings of the 3rd Workshop on e-Commerce and NLP*, pages 64–68, Seattle, WA, USA. Association for Computational Linguistics.

Shansan Gong, Zelin Zhou, Shuo Wang, Fengjiao Chen, Xiujie Song, Xuezhi Cao, Yunsen Xian, and Kenny Zhu. 2023. Transferable and efficient: Unifying dynamic multi-domain product categorization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 476–486, Toronto, Canada. Association for Computational Linguistics.

Sylvain Goumy and Mohamed-Amine Mejri. 2018. Ecommerce product title classification. In *Proceedings of the Workshop on eCommerce (co-located with SIGIR)*.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Vivek Gupta, Harish Karnick, Ashendra Bansal, and Pradhuman Jhala. 2016. Product classification in E-commerce using distributional semantics. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 536–546, Osaka, Japan. The COLING 2016 Organizing Committee.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing.

Ke Ji, Yixin Lian, Jingsheng Gao, and Baoyuan Wang. 2023. Hierarchical verbalizer for few-shot hierarchical text classification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2918–2933, Toronto, Canada. Association for Computational Linguistics.

Yugang Jia, Xin Wang, Hanqing Cao, Boshu Ru, and Tianzhong Yang. 2018. An empirical study of using an ensemble model in e-commerce taxonomy classification challenge. In *Proceedings of the Workshop on eCommerce (co-located with SIGIR)*.

Fei Li, Yonghao Jin, Weisong Liu, Bhanu Pratap Singh Rawat, Pengshan Cai, Hong Yu, et al. 2019. Fine-tuning bidirectional encoder representations from transformers (bert)–based models on large-scale electronic health record notes: an empirical study. *JMIR medical informatics*, 7(3).

Maggie Yundi Li, Liling Tan, Stanley Kok, and Ewa Szymanska. 2018. Unconstrained product categorization with sequence-to-sequence models. In *Proceedings of the Workshop on eCommerce (co-located with SIGIR)*.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. 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 188–197, Hong Kong, China. Association for Computational Linguistics.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy. Association for Computational Linguistics.

Carlos N Silla and Alex A Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22:31–72.

Michael Skinner. 2018. Product categorization with LSTMs and balanced pooling views. In *Proceedings of the Workshop on eCommerce (co-located with SIGIR)*.

Shogo D. Suzuki, Yohei Iseki, Hiroaki Shiino, Hongwei Zhang, Aya Iwamoto, and Fumihiko Takahashi. 2018. Convolutional neural network and bidirectional lstm based taxonomy classification using external dataset at sigir ecom data challenge. In *Proceedings of the Workshop on eCommerce (co-located with SIGIR)*.

Zihan Wang, Peiyi Wang, Tianyu Liu, Binghuai Lin, Yunbo Cao, Zhifang Sui, and Houfeng Wang. 2022. HPT: Hierarchy-aware prompt tuning for hierarchical text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3740–3751, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Li Yang, E Shijia, Xu Shiyao, and Xiang Yang. 2020. Bert with dynamic masked softmax and pseudo labeling for hierarchical product classification. In *Proceedings of the semantic web challenge on mining the web of HTML-embedded product data co-located with the 19th international semantic web conference*.

Wenhu Yu, Zhiqiang Sun, Haifeng Liu, Zhipeng Li, and Zhitong Zheng. 2018. Multi-level deep learning based e-commerce product categorization. In *Proceedings of the Workshop on eCommerce (co-located with SIGIR)*.

Wen Zhang, Yanbin Lu, Bella Dubrov, Zhi Xu, Shang Shang, and Emilio Maldonado. 2021. Deep hierarchical product classification based on pre-trained multilingual knowledge. *IEEE Data Eng. Bull.*, 44(2):26–37.

DAP-LeR-DAug: Techniques for enhanced Online Sexism Detection

Jayant Panwar
LTRC, International Institute of
Information Technology, Hyderabad
jayant.panwar@research.iiit.ac.in

Radhika Mamidi
LTRC, International Institute of
Information Technology, Hyderabad
radhika.mamidi@iiit.ac.in

Abstract

The swift surge of digital communication on social media platforms has brought about an increase in hate speech online, especially sexism. Such content can have devastating effects on the psychological well-being of the users, and it becomes imperative to design automated systems that can identify and flag such harmful content. Human moderation alone is inadequate to manage the volume of content, necessitating efficient technological solutions. In this study, we explore the performance of different modern techniques on Bert-based models for detecting sexist text. We explore four such techniques, namely, Domain Adaptive Pre-training (DAP), Learning Rate Scheduling (LeR), Data Augmentation (DAug), and an ensemble of all three. The results show that each technique improves performance differently on each task due to their different approaches, which may be suited to a certain problem more. The ensemble model performs the best in all three subtasks. These models are trained on a Semeval'23 shared task dataset, which includes both sexist and non-sexist texts. All in all, this study explores the potential of DAP-LeR-DAug techniques in detecting sexist content. The results of this study highlight the strengths and weaknesses of the three different techniques with respect to each subtask. The results of this study will be useful for researchers and developers interested in developing systems for identifying and flagging online hate speech.

1 Introduction

Text classification tasks have been around for a long time, and so has online hate speech. Posting without any consequences is stimulus enough for people to be overly hurtful in their comments and be ignorant of others' feelings. Some might just do it to "troll" someone, some out of pure hatred, and some for channelling their inner frustration. With time, the presence of hate speech prevalent online increases too, and all the major social platforms nowadays are trying to find ways to flag and curb it. Sexism has been present since before the Internet, and thus, there is no surprise that it is one of the most used forms of hate speech online today.

In our study, we aim to develop an automated system that can detect and classify sexism using different techniques, namely, Domain Adaptive Pre-training (DAP), Learning Rate Scheduling (LeR), and Data Augmentation (DAug). For the same, we use the dataset shared by the task organizers of Task-10 of SemEval-2023 (Kirk et al., 2023). The dataset contains data for the following three subtasks:

- **Subtask-A**: binary classification task in which systems must figure out whether a certain piece of text is sexist or not
- **Subtask-B**: systems must classify the sexist piece of content into its appropriate class from the given 4 classes
- **Subtask-C**: systems must accurately classify the sexist text into one of the listed 11 classes

Further details regarding sexism category names can be seen in Figure-1. As visible from the definitions discussed above, the complexity of the task increases with each level. We go from dealing with a simple binary classification task to an 11-class multi-classification problem. This is precisely why we tackle the task with three unique techniques and an ensemble of all three combined techniques. For implementing the these techniques we use three BERT-based models, namely, RoBERTa, HateBERT, and BERTweet. The best model for each task is the ensemble model. This is because each of the three techniques is beneficial in its own way and using an ensemble model makes sure that the advantages of all three techniques are utilized simultaneously.

DAP boosts the scores most for Task-A, LeR for Task-B, and DAug for Task-C thanks to their

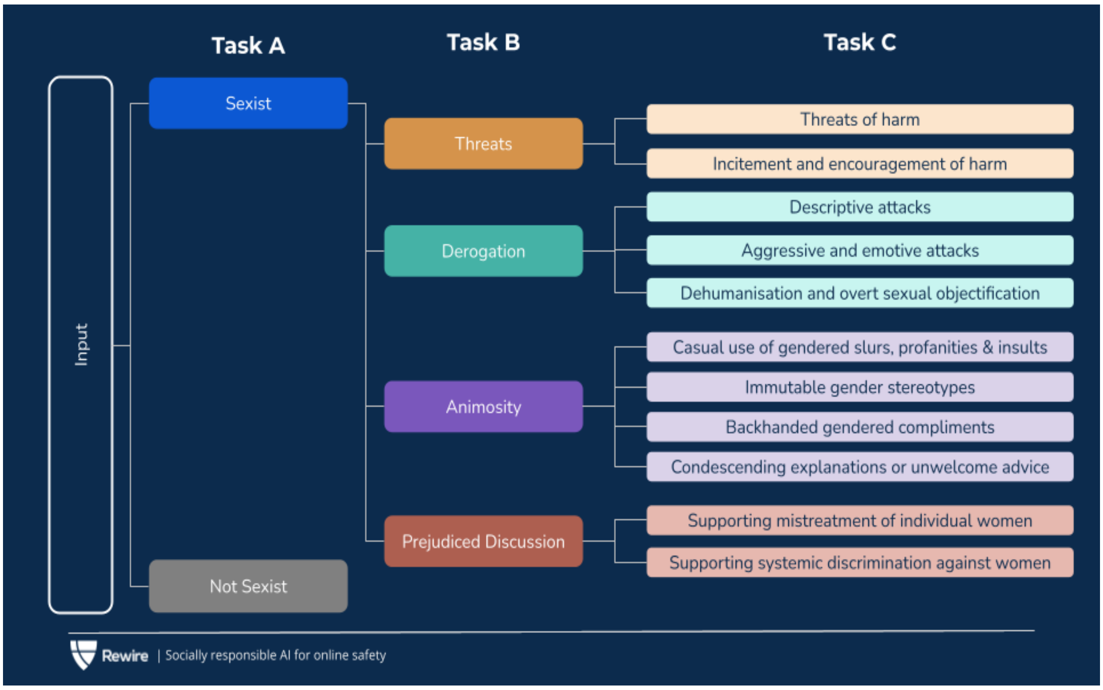

Task A
Task B
Task C
Input
Sexist
Not Sexist
Threats
Derogation
Animosity
Prejudiced Discussion
Threats of harm
Incitement and encouragement of harm
Descriptive attacks
Aggressive and emotive attacks
Dehumanisation and overt sexual objectification
Casual use of gendered slurs, profanities & insults
Immutable gender stereotypes
Backhanded gendered compliments
Condescending explanations or unwelcome advice
Supporting mistreatment of individual women
Supporting systemic discrimination against women
Rewire | Socially responsible AI for online safety

Figure 1: The shared task categories. Image adopted from the organizers of Kirk et al. (2023)

unique approach which caters to the respective subtasks. As a result of their ensemble model, our system is comfortably able to beat the best baseline model from the original task paper (Kirk et al., 2023).

2 Related Work

Detection of online sexism has been a task that many researchers have worked on over the past many years. Some showed how we can use both conventional and deep learning approaches to identify various forms of sexism in a multi-lingual setting (Rodríguez-Sánchez et al., 2020) while others have created their own datasets to examine different forms of sexist content prevalent nowadays (see Parikh et al. (2019), Samory et al. (2021)). In our study, however, we stick to the dataset for the EDOS task, so we can compare the performance of our systems with other major baselines and top-ranked systems.

There have also been important efforts when it comes to adapting the models to a certain domain. In our case that is adapting BERT-based models (for BERT see Devlin et al. (2019)) to hate speech, sexism to be specific. The authors of Gururangan et al. (2020) have shown how models can improve in performance by adapting a certain domain. For this, first, the model is trained on a large unlabelled dataset and then fine-tuned on the smaller labelled dataset, which fits in line with our case. This is where the motivation of the DAP technique comes from.

Zhao et al. (2022) showcased how important it is for the learning rates to adapt to the task so as to achieve best performance in classification tasks. This helps in faster convergence while training which ultimately leads to better results. Similarly, Data augmentation has always been shown to improve performance generally in text classification tasks. For instance, the EDA framework (Wei et al., 2019), where simple updates like synonym replacement, random insertion, random swap, and random deletion improved classification performance by a good extent. Likewise, there are other data augmentation approaches such as stochastic replacement of words in the sentence (Kobayashi, 2018), and using Pre-trained Language Models to get diverse and semantically correct text samples (Anaby-Tavor et al., 2019). In our study, we choose to stick with the simpler EDA approach.

3 System Overview

The system for our study can be broken up into 5 different parts. Firstly, we have the Bert-based models as it is, i.e., we do not employ any techniques on them. Then, we have got our DAP-LeR-DAug individual models to understand which technique works best in which scenarios. Finally, we wrap it all up by having an X model, which is basically an ensemble of all the three techniques discussed.

3.1 Baselines

As mentioned earlier, we will have three BERT-based models as our baselines, namely, RoBERTa, HateBERT, and BERTweet. RoBERTa (Liu et al., 2019) is an advanced BERT-based pretraining approach that optimizes and enhances performance on various natural language understanding tasks through extensive training with larger batches and more data, resulting in improved language representations. HateBERT (Caselli et al., 2021), on the other hand, is a specialized transformer-based model tailored for detecting hate speech in text, designed to provide accurate identification of offensive content through fine-tuned representations and focused training on hate speech data. Finally, BERTweet (Nguyen et al., 2020) is an adaptation of the BERT model specifically designed for processing and understanding text from social media platforms like Twitter, offering improved performance on tasks involving informal language, hashtags, mentions, and other characteristics unique to Twitter discourse.

It is evident from the description of the selected BERT-based models as to why they are apt for our experiment which is heavily focused on natural language understanding and dealing with sexism, a form of hate speech. For the baseline stage, we use them as they are and fine-tune them on our shared task dataset. Then we evaluate how they perform.

3.2 DAP

DAP refers to Domain Adaptive Pre-training. The organizers of the task (Kirk et al., 2023) had also provided a dataset of 2 million unlabelled posts from Gab and Reddit. We utilize this enormous dataset with the Masked Language Modelling (MLM) objective as we believe this pairing would hold the most promise for enhancing the performance of our BERT-based models in classifying sexist content. By being subjected to diverse and extensive linguistic contexts from the unlabelled dataset during MLM pretraining, the models gain a robust understanding of general language patterns and nuances. This enriched linguistic foundation forms the cornerstone for improved comprehension of text, enabling the models to capture subtle linguistic cues and contextual variations inherent in sexist content.

During fine-tuning with labelled data, the models' already adept language representations are seamlessly adapted to the specific domain of sexism detection. This dual-stage process harmonizes its universal language understanding with domain-specific features, resulting in heightened discriminatory power to accurately identify and classify sexist text instances. The fusion of pretraining's broad language expertise and fine-tuning's task-specific tailoring equips the models with a well-rounded ability to identify and categorize nuanced and varied forms of sexist content across the different classes of sexist content.

3.3 LeR

LeR refers to Learning Rate Scheduling. Learning rate scheduling enhances model performance by dynamically adjusting the step size during training. This technique accelerates convergence by initially allowing larger parameter updates, ensuring quicker progress towards the optimal solution. As training advances, the learning rate is reduced, stabilizing optimization and preventing overshooting. By navigating the loss landscape more effectively, learning rate scheduling helps evade local minima and improves generalization by mitigating noise fitting. Although this technique does not contribute linguistically in terms of word embeddings, contextual understanding of the domain, etc., it can still prove to be very important.

This technique is particularly valuable for stabilizing training with large batch sizes, adapting to data characteristics, and achieving fine-tuned results in transfer learning scenarios. In essence, learning rate scheduling fine-tunes the learning process itself, fostering quicker convergence, robustness, and overall improved model performance.

3.4 DAug

DAug implies Data Augmentation. The dataset we have is highly imbalanced for each subtask. For example, the majority class in tasks A and B has more than 3 times the number of data instances as compared to the minority class. For task C, the case is even worse. There are minority classes with not even 100 instances while some majority classes have more than 700 instances. A dataset like this can make the best of classifying models biased towards the majority class. There are various different techniques to counter that, and Data augmentation is certainly one of them. It concerns itself with creating new data for classes with lim-

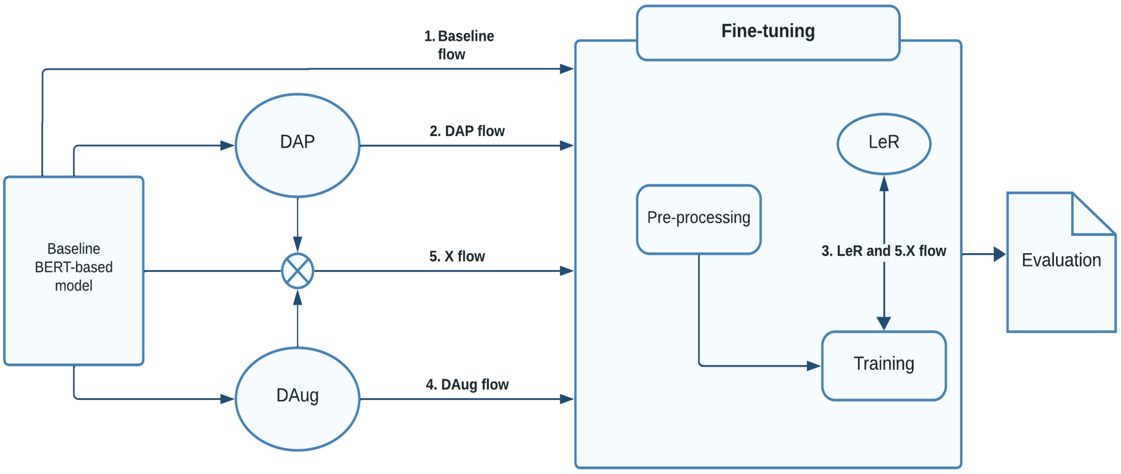


Figure 2: System Architecture

ited data available. It can significantly enhance a dataset with limited sexist posts by generating diverse variations of existing examples. Techniques such as synonym replacement, paraphrasing, and introducing minor textual perturbations help create additional instances of sexist content. By simulating different linguistic expressions and contexts, data augmentation enriches the dataset, which can, in turn, improve the model generalization and performance, even when original sexist instances are sparse.

As discussed in earlier sections, we make use of a similar approach as taken by authors of Easy Data Augmentation (EDA) (Wei et al., 2019). By introducing synonym replacement, random insertion, random swap, and random deletion to the text, the EDA framework generates diverse instances of the original text. This augmented data enriches the training dataset, improving model generalization and performance. EDA is demonstrated to be remarkably effective across various text classification tasks, showcasing its ability to alleviate the challenges posed by limited training data and contributing to more robust and accurate text classification models. This is why the EDA approach will be helpful for us, for all three subtasks. We discuss the exact setup details in the coming section.

3.5 X

The final or the X part of our system is basically a combination or an ensemble of all the three unique techniques we have discussed thus far. The ensemble capitalizes on the complementary strengths of each technique, effectively navigating linguistic complexities through pre-trained domain understanding, fine-tuning with task-specific context, and enriched data diversity. This holistic approach promotes greater robustness to nuances in sexist content and addresses challenges posed by limited labelled data. Ideally, this should outperform the individual technique models and ultimately lead to the best performance when it comes to classifying sexist content.

4 Experimental Setup

We discuss our experimental setup (see Figure-2) in two forms: technique and fine-tuning specific. Fine-tuning specific setup is applied to all the five models irrespective of the technique being used. We discuss the LeR setup in technique specific section, but we must remember that it is applied only while fine-tuning.

4.1 Technique specific

As discussed beforehand, one of the major problems we have is the class imbalance in the dataset. For that, we use the Data Augmentation technique. But, in order to do justice to other techniques so as not to make their classifiers biased toward the majority class, we had to consider other approaches for them like Undersampling and Oversampling. In Undersampling, we remove a certain number of data instances from the majority class to make sure the classes are more or less balanced. However, in Oversampling, we do the opposite. We replicate data instances of the minority class until we have achieved balance among all the classes in the dataset. Undersampling has been shown to perform better for this shared task (Panwar and Mamidi, 2023), while Oversampling has been shown to perform worse than using the dataset as it is, i.e., imbalanced. Therefore, for the model variations that do not include data augmentation, i.e., Baseline, DAP, and LeR, we use Undersampling to balance the dataset.

Regarding the setup for Data Augmentation models, we implement the EDA framework (Wei et al., 2019), as explained earlier. For generating data, we decided to choose RoBERTa as it gave semantically closer data to the actual data when compared with the data generated by HateBERT and BERTweet. We limit the augmentation probability to 0.3 as above this threshold, the system generates very noisy data, which can lead to loss of semantics and an overall reduction in the performance of the models.

For the Domain Adaptive Pre-training technique, we use the Masked Language Modelling objective. The first and foremost step is to obviously use the correct tokenizers and pre-process tokens that may not contribute semantically a lot to the sentence. For example, tokens like [USER], [HASHTAGS], [URLS], [MENTIONS], etc can be removed to improve efficiency and accuracy. Then we create the masked sentences, and we do so by randomly masking a certain percentage of the sentence. Then, the model learns by predicting the masked tokens based on the surrounding context. The goal is to minimize the loss between the actual masked and predicted tokens. By gaining a better idea of the contextual relationships from posts on sexist forums, the model should ideally perform better than without DAP.

For the Learning Rate scheduling models, we experimented primarily with four different types of LRs: Step decay, Exponential decay, Cosine annealing, and One-Cycle LR. They performed more or less similarly, with the only difference being when it came to the X or the ensemble model. In that case, cosine annealing edges out other approaches and this may be due to the fact that the X model has a lot going underneath the layers. Not only does it have more contextual embeddings thanks to DAP, but it also has more data to work with because of DAug. These rising complexities require complex learning rate scheduling policies like that of Cosine Annealing.

4.2 Fine-tuning specific

This part is very intuitive. We split the dataset into 85:15 ratio with the former used for training and the latter for validation. The authors of the task have provided separate data for testing and we believe it would be better to test our models on that to compare how we stand with task paper baselines and other top-ranked teams. During the training phase, first, we do simple pre-processing. Most of the pre-processing is handled comfortably with the appropriate tokenizers of the different models we have considered. However, we take care extra care on our own end to remove tokens that do not contribute semantically to the system. For example, hashtags, emojis, noisy tokens like "heyyyyyy", "yolooooooo", etc. For training our classifiers, we set epochs as 10 and batch size as 16. After training the classifiers, we proceed to evaluate them.

5 Results

For evaluation, we make use of macro average F-1 scores. This helps us to compare the performance of our approach with that of the task paper baselines and other top-ranked teams. A major reason for adopting macro average F-1 scores could be that during evaluation it treats each class of the dataset appropriately. This is very beneficial in cases, where the dataset is highly imbalanced, like in our case.

From the results in Table-1, we can see that the ensemble model with RoBERTa as baseline performed the best on the evaluation test. There can be different reasons for that, but the primary reason has to be the architecture of RoBERTa and the fact that we have used RoBERTa-base in our Data Augmentation phase. Models like HateBERT and BERTweet have a good understanding of hate speech beforehand, thanks to their architecture and pre-training. It is possible that techniques like DAP and DAug did not help these models as much as they helped RoBERTa since they have been exposed to a wide variety of hate speech data and our techniques did not increase their contextual understanding or vocabulary a whole lot.

Another important point to note is that the X model performs the best for each baseline. All three techniques that we decided upon, when employed together, can cause the model to perform best. It is also intuitive as the X model is one which has been pre-trained heavily on about 2 million posts for adapting the sexist content domain,

Model	*Task-A: 2 class*	*Task-B: 4 class*	*Task-C: 11 class*
RoBERTa-base	83.22	59.77	34.01
RoBERTa-DAP	84.67	62.23	36.44
RoBERTa-LeR	82.45	63.97	38.21
RoBERTa-DAug	83.59	63.87	39.89
RoBERTa-X	**85.09**	**65.89**	**40.23**
HateBERT-base	82.13	60.56	33.84
HateBERT-DAP	82.55	62.23	35.19
HateBERT-LeR	82.14	64.12	37.77
HateBERT-DAug	82.34	64.01	38.09
HateBERT-X	82.78	64.53	38.34
BERTweet-base	84.01	61.12	30.01
BERTweet-DAP	84.33	62.01	32.71
BERTweet-LeR	84.03	62.89	34.66
BERTweet-DAug	84.12	62.88	34.81
BERTweet-X	84.39	63.45	35.22

Table 1: Macro Avg. F-1 Scores of Classifiers on all subtasks

has got augmented data with minority classes also being represented adequately, and finally, can train optimally thanks to the learning rate scheduling technique. The three techniques complement each other and bring out the best when used together.

We really notice the impact of individual techniques when we look at the results task-wise. For task A, we can see that the DAP technique improves the score the most on the baseline. This is intuitive as well because for a simple binary classification subtask, having more embeddings and wider vocabulary to work with makes it even easier for the model to figure out if the content is plain sexist or not. The LeR approach works best with increasing complexities of the task. It works better for Tasks B and C than it does for Task A. The effect of optimal convergence is noticed more easily when there are more classes involved in the task. It performs the best for task B and is also good for task C. It is not that its performance drops in task C but that Data augmentation works too well for task C and it outshines the LeR technique. We have established multiple times in this study that the dataset is imbalanced, and this imbalance increases with the increasing complexity of the task. Undersampling can only work so well when we have to deal with 11 classes in task C, and the majority of them are very under-represented. This is where Data Augmentation comes in handy. By creating more data instances for the minority classes, we are able to give the model more data to work with and thus increase its performance in classification.

Model	*Task A*	*Task B*	*Task C*
Best Baseline	82.35	59.26	31.71
Top-ranked	87.46	73.26	56.06
RoBERTa-X	85.09	65.89	40.23

Table 2: Comparison of the performances of the Best Baseline model in Task paper, the top-ranked systems for each subtask, and our best performing model: RoBERTa-X

Lastly, we compare our best-performing model, i.e., RoBERTa-X, with the best baseline model of the task paper (Kirk et al., 2023) and the top-ranked systems for each subtask. We are able to comfortably beat the best baseline model in each of the subtasks, thanks to the ensemble of our effective techniques. We were not able to beat the top-ranked system in any subtask, even though we came close. However, we must note that for this shared task, no single approach was the top-ranked among all the three subtasks. The top-ranked system score for each subtask in Table-2 is from a different team. We were able to create a single approach that at least beat the best baseline. Comparing our scores with the task leaderboard, we would stand in the top 30% submissions in task A, top 25% submissions in task B, and top 40% submissions in task C.

6 Conclusion

Through this study, we were able to explore the effectiveness of the DAP-LeR-DAug techniques when it comes to classifying hate speech in the form of sexism. We were able to demonstrate that each technique works well with a specific subtask, and when employed together in the form of an ensemble, they perform the best, irrespective of the BERT-based model being used. This goes on to show that the scores achieved were not coincidental, and the techniques indeed complement each other in a good way.

Although the DAP-LeR-DAug techniques do not perform the best for any specific subtask when compared with top-ranked systems, it should be pointed out that they do surpass the scores achieved by the best baseline model in the original task paper quite comfortably. Nevertheless, there are a lot of ways to improve upon the scores achieved, which we discuss in the next section.

Limitations

Like any other research study, ours, too, is filled with limitations. Overcoming some of these would directly result in better scores for each subtask while some others may increase the training time but nonetheless will improve the performance of the models.

First of all, we have used only the base versions of the BERT-based models. If not for the restraint of computational resources, we could have used the large, extra-large, versions of the baseline models. The larger vocabulary and increased number of parameters would directly help to achieve better scores in all three subtasks.

Another way to improve our performance could be using more data for DAP. The suggestion is indeed greedy but will improve the performance nonetheless. Similarly, we could experiment with other forms of hyperparameter tuning apart from LeR alone. Some of them could be optimizing the dropout rate, loss functions, weight decay, and activation functions. The impact of tuning these may not be very large but it will optimize our performance.

We can also try to use different data augmentation approaches. In our study, we have only used the EDA approach but there are more complex ways to augment data. For example, Back-translation, in which we translate the English sentence to a certain language and then back to English. This is an easy and effective way to generate more samples for under-represented classes and ultimately balance the dataset.

Lastly, we can try to improve our pre-processing stage as well. In our pre-processing stage, we get rid of all the emojis and hashtags but they have been shown to improve the performance of classification tasks (Eisner et al., 2016). They can be converted to vector embeddings and then combined with our word embeddings to form custom vector embeddings. This will directly improve the performance of our model as emojis are used a lot on social platforms nowadays and they contribute to the context and semantics of the text.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2019. Not enough data? deep learning to the rescue! *Computing Research Repository*, arXiv:1911.03118. Version 2.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for abusive language detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, 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.

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description. In *Proceedings of the Fourth International Workshop on Natural Language Processing for Social Media*, pages 48–54, Austin, TX, USA. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasovi'c, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *Computing Research Repository*, arXiv:2004.10964. Version 3.

Hannah Rose Kirk, Wenjie Yin, Bertie Vidgen, and Paul Röttger. 2023. SemEval-2023 Task 10: Explainable Detection of Online Sexism. In *Proceedings of the 17th International Workshop on Semantic Evaluation*, Toronto, Canada. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. 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 452–457, New Orleans, Louisiana. 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. *Computing Research Repository*, arXiv:1907.11692. Version 1.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, Online. Association for Computational Linguistics.

Jayant Panwar and Radhika Mamidi. 2023. PanwarJayant at SemEval-2023 task 10: Exploring the effectiveness of conventional machine learning techniques for online sexism detection. In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 1531–1536, Toronto, Canada. Association for Computational Linguistics.

Pulkit Parikh, Harika Abburi, Pinkesh Badjatiya, Radhika Krishnan, Niyati Chhaya, Manish Gupta, and Vasudeva Varma. 2019. Multi-label categorization of accounts of sexism using a neural framework. 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 1642–1652, Hong Kong, China. Association for Computational Linguistics.

Francisco Rodríguez-Sánchez, Jorge Carrillo-de Albornoz, and Laura Plaza. 2020. Automatic classification of sexism in social networks: An empirical study on twitter data. *IEEE Access*, 8:219563–219576.

Mattia Samory, Indira Sen, Julian Kohne, Fabian Floeck, and Claudia Wagner. 2021. "call me sexist, but...": Revisiting sexism detection using psychological scales and adversarial samples. *Computing Research Repository*, arXiv:2004.12764. Version 2.

Jason Wei, Kaiqing Zou, Mingxuan Chen, and Lei Li. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *Computing Research Repository*, arXiv:1901.11196. Version 2.

Kailin Zhao, Xiaolong Jin, Saiping Guan, Jiafeng Guo, and Xueqi Cheng. 2022. MetaSLRCL: A self-adaptive learning rate and curriculum learning based framework for few-shot text classification. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2065–2074, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

CommunityFish:
A Poisson-based Document Scaling With Hierarchical Clustering

Sami Diaf
Universität Hamburg
Department of Socioeconomics
sami.diaf@uni-hamburg.de

Abstract

Document scaling has been a key component of modern text-as-data applications in social sciences, particularly for political scientists, who aim at uncovering differences between speakers or parties with the help of probabilistic and non-probabilistic approaches. Yet, most of these techniques employ the bag-of-word hypothesis and disregard semantic features or use prior information borrowed from external sources that may bias the results. This paper presents *CommunityFish* as an augmented version of *Wordfish* based on a prior hierarchical clustering of the word space to retrieve semantic n-grams, or *communities*, as signals emerging from the corpus to be used as an input to *Wordfish*. Instead of considering all words in the corpus as independent features, we emphasize the interpretability of the results, since communities have the ability to better scale parties or speakers, and ensure a faster convergence when considering a Poisson-based ranking model. Aside from yielding communities assumed to be subtopics summarizing the corpus' narrative signals, the application of this technique outperforms the classic *Wordfish* model by emphasizing key historical developments in the U.S. State of the Union addresses and was found to replicate the prevailing political stance in Germany when using the corpus of parties' manifestos.

1 Introduction

Comparative politics has been a prominent domain of application of what is currently known as text-as-data field, featuring the use of text mining techniques and machine learning algorithms to identify patterns that differentiate documents or track disparities at the meta-data level. Scaling techniques typically comprise an array of unsupervised methods, both probabilistic and non-probabilistic, which aim to extract one or multiple dimensions to enable metadata comparisons, based on a set of assumptions conducted at the word-level.

Earlier scaling techniques used statistical learning approaches as for matrix factorization schemes (Deerwester et al., 1990) and a probabilistic model based on the Poisson distribution as for *Wordfish* (Slapin and Proksch, 2008; Lowe and Benoit, 2013) which ranks documents on a unidimensional scale using word occurrences in the corpus. Further extensions of Poisson scaling models considered a debate structure (Lauderdale and Herzog, 2016), pre-trained embedding models (Nanni et al., 2019), word variations (Vafa et al., 2020) and semantic search strategies (Diaf and Fritsche, 2022b), providing an improved scaling of documents depending on several assumptions and use cases at the word or document levels.

Regarding *Wordfish*, the Poisson scaling model uses word counts to learn a hidden and normally-distributed dimension, assumed to be a proxy of partisanship among political parties when scaling manifestos (Slapin and Proksch, 2008). However, the Poisson distribution does not always pertain (Lowe and Benoit, 2013), as frequent words are likely to be normally distributed, while very rare words tend to substantially deviate from the Poisson paradigm (Lo et al., 2016). Another disadvantage is the dynamic word usage which needs time-varying parameters for the Poisson ranking model and further constraints on parameters to ensure its stability (Jentsch et al., 2020), or to consider the structure *document-topic-word* to get polarization at the topic level using a hybrid supervised topic model (Diaf and Fritsche, 2022a).

Although the choice of scaling techniques is abundant, it may not always meet the expectation of practitioners, as the inference is done at the word-level, while the analysis often targets documents' content in terms of groups of words that convey the interest of researchers. The word contribution to the built scale in *Wordfish* is static and cannot be fully interpretable if the corpus has undergone significant changes over time, in terms of word usage, between parties/speakers (Jentsch et al., 2020). Furthermore, the polarity of specific words could be different from the position of documents they are mostly related to, thus not in-line with experts' assessments (Hjorth et al., 2015). This issue arises from the bag-of-word assumption and the underlying agnostic hypothesis of word independence, which prevents an accurate scaling of documents based on semantic features (Nanni et al., 2019).

Advances in social network analysis indicated that hierarchical clustering can reveal homogeneous and distinct groups of users, commonly referred to as *communities*, based on their interactions, which could also be used in text mining to identify independent, semantic groups of words, in form of n-grams, that differentiate documents by their occurrences while delivering informative signals that outperform analyses based on single-word usage. One popular algorithm for studying social networks is the *Louvain* algorithm (Blondel et al., 2008) which was applied to get word groups that better represent the rhetoric used in a given corpus (Bail, 2016) or to study the lexical shift in the State Of The Union addresses (Rule et al., 2015). Other hierarchical clustering schemes were proposed as for *Infomap* (Rosvall and Bergstrom, 2008) which uses random walk map-equation instead of optimizing the modularity as for *Louvain* (Lancichinetti and Fortunato, 2009), and *Leiden* (Traag et al., 2019) which was found to outperform *Louvain* when applied to big networks, however, similar performances with *Louvain* are expected on smaller networks.

This paper extends the idea of *lexical shift* (Rule et al., 2015) by identifying communities as representative groups of words, able to achieve a fast and interpretable scaling of documents upon which a Poisson ranking model could be built, instead of considering a plain word-count model related to the bag-of-word hypothesis. I argue that communities offer a better polarization level when differentiating documents and metadata than standard bag-of-word techniques, in addition to efficiently speeding up the learning process by reducing the size of the document-term-matrix whose sparsity may hinder the convergence of Poisson models. Commonly used words are likely to form communities with a high frequency of words but are less likely to be polarized compared to communities with exclusive word usage, denoting the focus of a given speaker/party on a specific subject of item that could be identified without the need to run topic models.

Two historical corpora, in English and German, were selected to evaluate this novel approach. The application on the U.S. State Of The Union (SOTU) addresses (1854-2019) shows a dominance of historical developments as for economic issues, local affairs and foreign policy that ranked addresses on a two-regime scale whose transition could be identified during the great depression. From the analysis of German political parties' manifestos (2013, 2017 and 2022), *CommunityFish* identified granular themes at the center of election debates that were found to replicate the ideological spectrum of political parties with *AFD* and *Linke* parties being the ideological bounds of the learned scale, while other parties seem to share many featured themes, hence reinforcing their centrist positions.

The paper outlines the build-up of *CommunityFish* from a network analysis perspective (Section 2) and from statistical learning (Section 3), then implements the proposed algorithm on two corpora (Section 4) and compares to the standard *Wordfish* used by practitioners.

2 Methodology

2.1 Network Analysis

Analysis of social media drove the attention of scientists on the necessity to adopt advanced clustering methods able to extract information that describe relationships between users via the types of messages or ideas they produce (White, 2008), instead of simple relationship structures between individuals (Bail, 2016).

Network analysis witnessed important contributions on identifying distinct subgroups in social networks, built on several optimization schemes developed to offer intuitive clustering (Lancichinetti and Fortunato, 2009).

For such tasks, researchers should carefully select clustering methods for community detection and also take into account centrality scores (Mester et al., 2021). *Louvain* algorithm (Blondel et al., 2008) is one commonly used clustering technique ,usually preferred to *FastGreedy* algorithm (Clauset et al., 2004), due to its relative low complexity, as it achieves a local optimization of the modularity Q at the node-level, defined as :

$$Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\delta(c_i, c_j)$$

with A_{ij} representing the edge weight between nodes i and j, k_i and k_j are the sum of the weights of the edges attached to nodes i and j, respectively; m is the sum of all of the edge weights in the graph; c_i and c_j are the communities of the nodes; and δ is Kronecker delta function $\delta(x, y) = 1$ if x=y, 0 otherwise.

Louvain clustering iteratively optimizes the modularity Q by starting with different node being its own community, and the concept is to place a node n_i to one of its neighboring nodes community, in a way to maximize the modularity change (Mester et al., 2021). Similar to users in social networks, *Louvain* algorithm can cluster words in a corpus, so to extract *communities*, in a form of n-grams of different lengths, having an independent, non-overlapping structure stemming from the specific word usage found in documents.

Traag et al. (2019) proposed *Leiden* clustering as a reliable alternative to *Louvain* in discerning small connected communities in large network structures. Altough *Leiden* was found to be faster than *Louvain*, in terms of execution, both do not differ when the network structure is relatively small, as for collection of documents with limited vocabulary, meaning the community structures of both algorithms can share many similarities and just slightly differ in the number of uncovered clusters.

2.2 Poisson ranking model

To apply *CommunityFish*, the corpus is broken down into bigrams and a minimum threshold π is set before running *Louvain* algorithm that yields K communities used as features for the Document-Term-Matrix (DTM), instead of considering all words in the corpus, hence communities serve as features to the Wordfish scaling algorithm. This scheme could be seen as a semantic clustering of the DTM that identifies correlated pairs of words in local contexts, thanks to a hierarchical clustering on bigrams, which differs from a simple bigram grouping of the initial DTM features.

The resulting DTM, as a matrix of communities' frequencies on each document in the corpus, is given as an input to *Worfish* (Slapin and Proksch, 2008) to learn document positions, or ideal points, that scale documents based on the occurrence of communities. As a scaling technique, *Wordfish* uncovers a latent scale θ, assumed to be a proxy of partisanship or ideological differences between parties or speakers, depending on the used context.

Although the use of Poisson distribution is justified by the occurrence of words in the corpus, assumed to be rare events, it is not always applicable to cases where the word usage concerns few documents, meaning the Poisson's expectation departs significantly from the variance (Lowe and Benoit, 2013; Lo et al., 2016) even though a quasi-Poisson scheme can relax the Poisson assumption of the mean-variance equality.

I argue that considering communities frees the DTM from potential biases raised by rare words and allows a faster convergence of *Wordfish* algorithm when applied to big corpora. *CommunityFish* could be seen as a double dimensionality reduction technique: first to uncover communities, as the primary unit of analysis, and second to learn one scale of ideal points using a Poisson ranking model.

Algorithm: CommunityFish

1.Community detection: Run a hierarchical algorithm (*Louvain*) over the bigram features of the corpus and extract K groups of words or *communities*, whose occurrence in the corpus is greater than π.

2.Poisson scaling model: The K communities are used as features for the Document-Term-Matrix, to be given as input to the Poisson scaling model (Slapin and Proksch, 2008) to uncover the scale θ_i from the specification:

$log(\lambda_{ij}) = \alpha_i + \psi_j + \theta_i\beta_j$, where:

λ_{ij}: frequency of the community j in document i

α_i: document fixed effect

ψ_j: community fixed effect

θ_i: the *position* of document i

β_j: the effect of community j to the document position

The hierarchical clustering applied to the corpus (*Louvain* algorithm) may be regarded as an implicit factorization of the traditional unigram DTM, yielding an interpretable feature matrix stemming from the learned communities. Aside from lowering the DTM dimension, it permits to intuitively concentrate the scaling on meaningful and independent groups of words (*communities*), that discriminate the ideal points based on their occurrences in the documents.

3 Application

3.1 State of the Union

State of the Union (SOTU) addresses consist of annual speeches given by U.S. presidents during the period (1854-2019), so to emphasize the duality democratic-republican in the scaling (Diaf and Fritsche, 2022a). The corpus was lemmatized using *udpipe* model (Straka et al., 2016) to reduce the size of the Document-Term-Matrix and learn robust communities, in comparison with the raw corpus. The application of the *Louvain* algorithm yielded 52 different communities (Table 1) with a clear historical context that spans over one and half century, tied to different episodes of modern American history. From Table 1, 22 communities, out of 52, are constituted of bigrams and the remaining are n-grams of different lengths comprising entities, expressions as well as plans or programs[1].

Communities, whose contributions to the scale β_j are different from zero, polarize the overall scale θ via their respective signs. From Figure 1, communities 45, 40, 11 and 8 contribute to documents whose positions in the overall scale (Figure 2) are positive, consisting of earlier addresses from the second half of the ninetieth century that targeted foreign policy and local administration. On the other hand, modern addresses have negative positions (Figure 2) and demonstrate a strong influence of foreign policy and defense interests (communities 38 and 49) as well as business/economic environment (communities 43 and 2). Figure 2 shows a two-regime scale of ideal points, whose transition occurred during the great depression (Hoover's addresses during the period 1929-1933, coinciding with the position $\hat{\theta} = 0$), suggesting a potential shift in the rhetoric, or a transition into modern addresses, used by U.S. presidents and captured via communities that could be assumed to be proxies for most discussed interests in their addresses.

In comparison to classic *Wordfish* application on the same corpus (Diaf and Fritsche, 2022a), the learned document positions are quiet similar, but cannot be differentiated in small periods, even if given by different speakers. Word contributions (Figure 5) obtained via *Wordfish* offer clustered, heavily centered densities, with tails dominated by rare words that occurred in a relatively small number of documents.

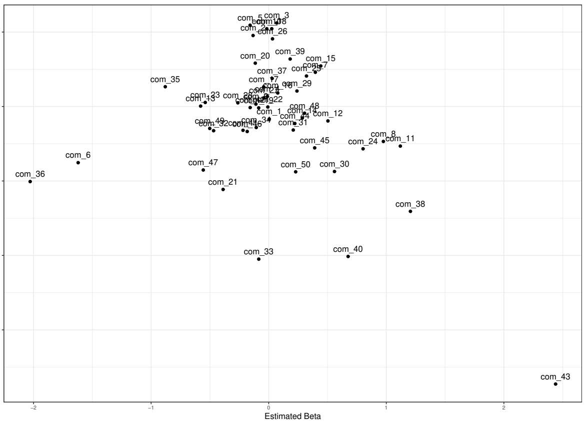


Figure 1: Communities contributions to the scale (β) vs communities' positions ψ (SOTU corpus)

3.2 German Manifesto

The corpus of Manifesto Project (Lehmann et al., 2022) was used to get the manifestos of six main German political parties, during the period 2013-2021 (Diaf and Fritsche, 2022b), then lemmatized using *udpipe* German language model (Straka et al., 2016) to reduce the vocabulary length of the corpus. It resulted 45 communities (Table 2) reproducing most of the debated themes in social life, politics and economic development which constitute the basis of the learned scale (Figure 4), found to replicate the prevailing political partisanship in Germany. The *AFD* and *Linke* parties represent the opposite ends of the learned scale, while the other parties hold central positions, with noticeable firm positions (small standard deviations of their ideal points) of the *Linke* and *Grüne* parties throughout the studied period. Conversely, the positions of *AFD* and *CDU* exhibit the highest variability, evidenced by wider standard errors. The blue line in Figure 4 is the local polynomial regression *Loess curve* (Jacoby, 2000) used to separate parties into two distinct classes (left-right) based on learned scale from the established communities (Table 2), resulting into a bi-partisanship *AFD-CDU-FDP* and *SPD-Grüne-Linke*.

From Figure 2, communities 40 and 45 support the position of the *Linke* party, as their contribution to the scale is strongly positive, in comparison to communities 5, 11 and 12 whose β_j are still positive but rather close to the origin. Most of the learned communities have a low contribution to the scale ($\beta_j \to 0$) and denote shared interests debated by political parties.

[1] *Leiden* clustering yielded a similar community structure to *Louvain*, with minor differences concerning two communities, out of 52. The same results were found using the German political manifesto corpus.

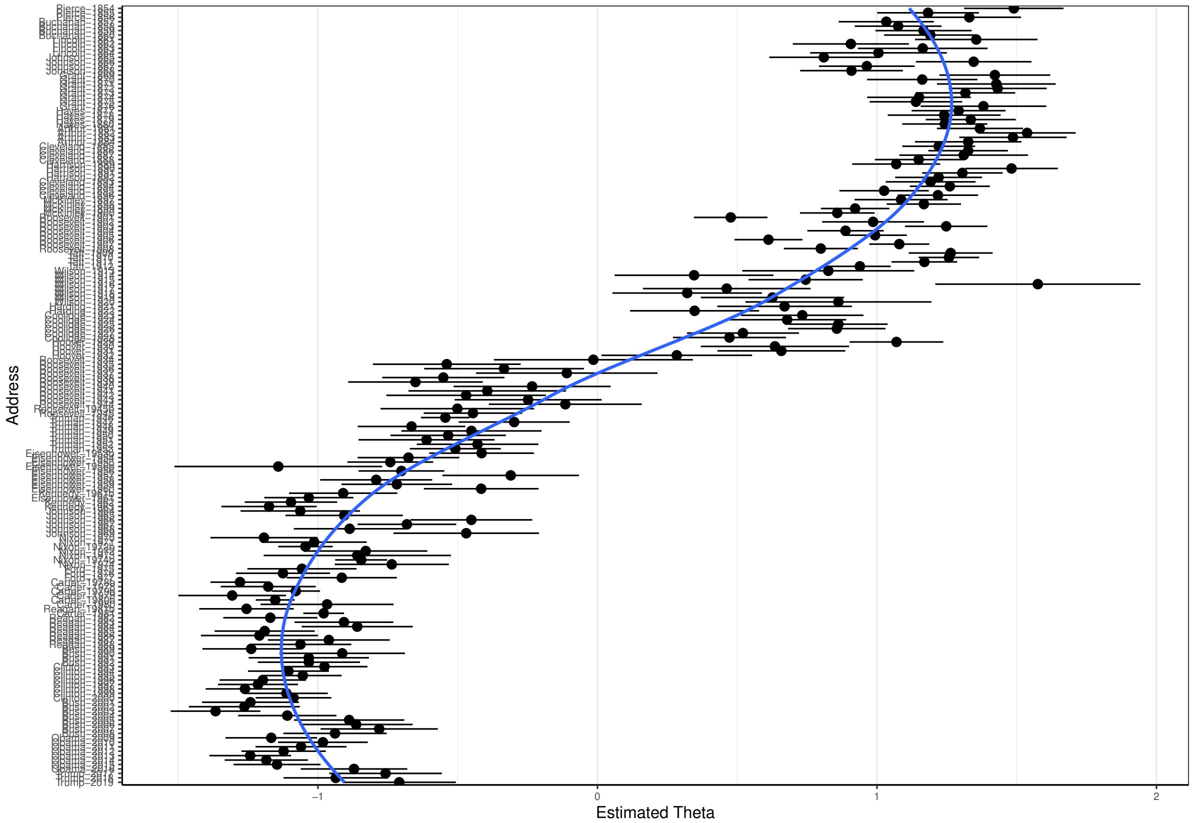


Figure 2: Learned CommunityFish ideal points with 95% confidence intervals (SOTU Corpus).

As a comparison to Wordfish (Figure 6), *CommunityFish* highlights a better polarization *AFD-Linke*, and a clear partisanship even if document positions exhibit a higher variability, in terms of standard errors, than *Wordfish*.

4 Conclusion

Scaling techniques are valuable analytical tools used by political scientists to explore partisanship among parties and to understand the ideological spectrum of speakers. Nonetheless, they are limited by the fact that they consider only words as the unit of analysis, making their application agnostic vis-à-vis semantic signals emerging from the corpus. While numerous solutions were developed to improve scaling results by incorporating external information sources as priors, the use of hierarchical clustering, as a pre-processing step, enables the identification of *communities*, as resilient clusters, with semantic effectiveness and substantial results, combined with a faster execution time. *CommunityFish* is a scaling technique that translates the unit of analysis from words to communities and an implicit factorization of the document-feature-matrix, unveiling informative sub-topic structures for an in-depth scaling of historical corpora as well as political manifestos. Optimal use of *CommunityFish* requires selecting most informative communities in an already-lemmatized corpus by mean of a clustering technique (such as *Louvain* or *Leiden* algorithms). This ensures an independent community structure when aggregating the document-feature-matrix, helping the spread of the ideological stance learned via Poisson ranking model, which was found to outperform classic *Wordfish* without calling expensive, often biased, prior information. Applied to two distinct corpora, it demonstrated a great ability in extracting communities from a language-variable corpus (SOTU) and identifying common items in debate-based documents (German manifesto) for an efficient and meaningful scaling of documents.

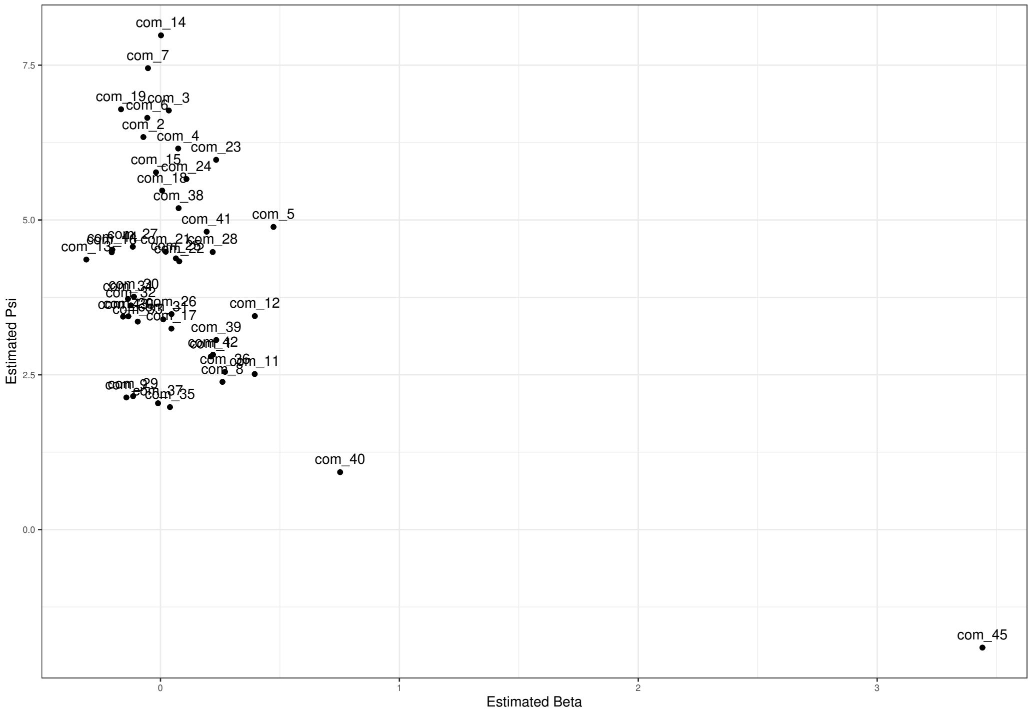


Figure 3: Communities contributions to the scale (β) vs communities' positions ψ (German Manifesto corpus)

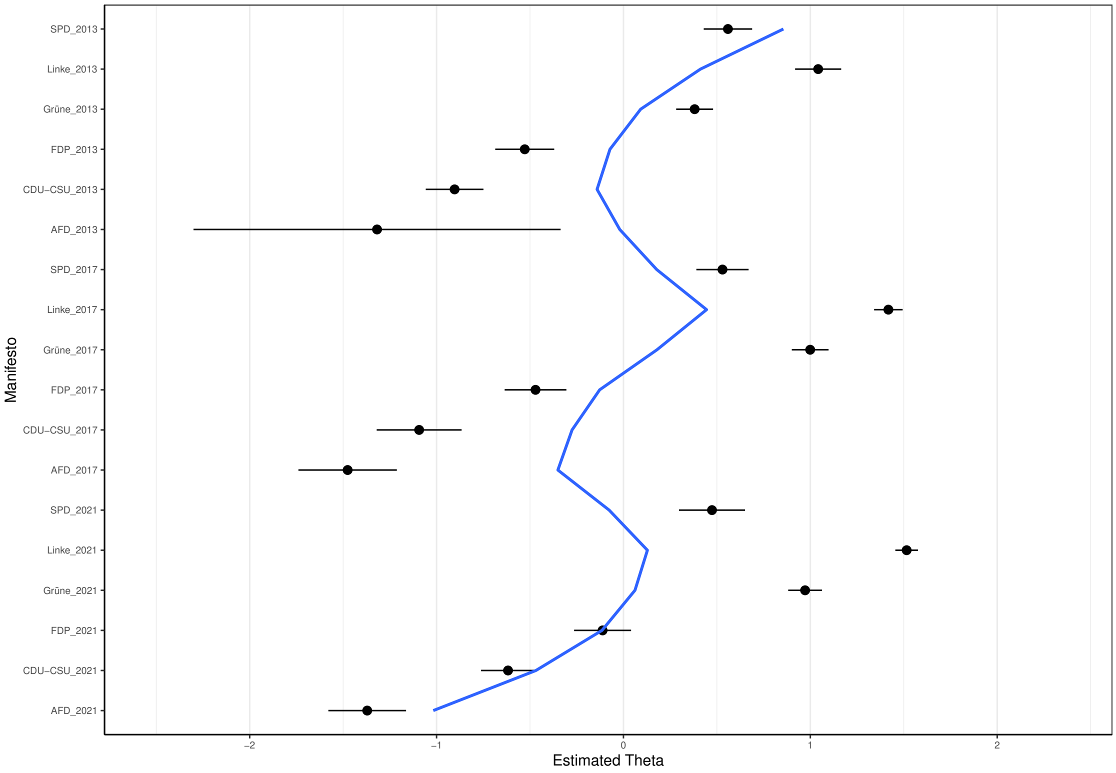


Figure 4: Learned CommunityFish ideal points with 95% confidence intervals (German Manifesto Corpus).

Table1: Communities in SOTU corpus

Community	Words
com_1	agricultural, product
com_2	american , billion , business , enlist , every , fellow , million , silver , small young , citizen , family , people , republics, dollar , man , day , americans
com_3	annual , special, message
com_4	armed , military, naval , force
com_5	ask , come , current , end , fiscal , five , four , last , many , next , past precede , previous, recent , ten , three , two , year , congress, june , session , ago , ahead
com_6	attorney , british , can , federal , general , government, local , make , must national , postmaster, self , social , spanish , supreme , help , court , sure also , continue , bank , defense , security
com_7	balanced, budget
com_8	base , call , confer , depend , enter , impose , urge , upon , attention
com_9	careful , favorable , consideration
com_10	central, latin , south , america
com_11	civil , hard , human , interest , postal , public , right , tax , work , service , rate debt , building , land , opinion , now , credit , cut , reduction, together
com_12	commerce , interstate, commission
com_13	earnestly, recommend
com_14	economic , development, growth
com_15	executive, branch , order
com_16	exist , international, law , present , tariff , enforcement , condition , system
com_17	far , thus , reach
com_18	first, time
com_19	foreign , free , great , nation , office , post , take , treasury , war , world country com_ , power , trade , britain , department, place , ii
com_20	full , employment
com_21	go , look , move , forward
com_22	god , bless
com_23	good , faith
com_24	health , medical , care , insurance
com_25	high , level , priority, school
com_26	internal, revenue
com_27	large , number, part
com_28	let, us
com_29	long, run , term
com_30	low , income
com_31	may , well
com_32	merchant, marine
com_33	middle, class , east
com_34	minimum, wage , worker
com_35	mr , speaker
com_36	natural , resource
com_37	new , job , program, york
com_38	nuclear, weapon
com_39	one , half , hundred, third
com_40	panama, canal
com_41	per , annum, cent
com_42	philippine, islands
com_43	private , enterprise, sector
com_44	progress, step , toward
com_45	puerto, rico
com_46	set , forth
com_47	several, united , states , nations
com_48	sink, fund
com_49	soviet, union
com_50	vice , president
com_51	welfare, reform
com_52	white, house

Table 2: Communities in German Manifesto corpus

Community	Words
com_1	abkomme, abkommen
com_2	afd, demokrat, deshalb , fordern , frei, linke, stehen, setzen
com_3	alt, brauchen, immer, jung, mehr , mensch million, gerechen, stark, geld, personal, transparenz, zeit
com_4	arbeit, beruflich, gut, kulturell, selbstbestimmt, arbeiten bildung, arbeitsbedingung, leben, zukunft
com_5	arbeitgeber , arbeitnehmer, patient , verbraucher , innen
com_6	arbeitsplatz , dass , deutschland , einsetzen , ganz , gestalten jed , neu , schaffen , sicherstellen, sorgen verhindern zeigen , einzeln , form , kind , technologie
com_7	beitrag , bund , dabei, etwa, gelten , gerade , gesellschaftlich, insbesondere , land , mittel, projekt , regelung sollen , sowie , teilhabe, wichtig, zugang , leisten, na, mitteln , rolle
com_8	bezahlbar, wohnraum
com_9	biologisch, vielfalt
com_10	cdu, csu
com_11	corona, krise
com_12	demokratisch, kontrolle
com_13	deutsch, bundestag, sprache
com_14	digital, it, sozial, infrastruktur, welt , sicherheit, absicherung gerechtigkeit, marktwirtschaft , netzwerk, sicherungssystem wohnungsbau, zusammenhalt
com_15	drei, euro, letzt, milliarde, mrd, pro, seit, vergangen, vier, zehn, jahr
com_16	erhalten, bleiben
com_17	erneuerbare , erneuerbaren, energie, energien
com_18	erst, schritt
com_19	eu, ebene, kommission, mitgliedstaat, staat
com_20	fair, wettbewerb
com_21	gering, hoch, mittler, einkommen , unternehmen
com_22	gesetzlich, mindestlohn, rent, rentenversicherung
com_23	gleich, recht , chance, lohn , rechte
com_24	hartz, iv
com_25	lage, versetzen
com_26	medizinisch, versorgung
com_27	nachhaltig, wirtschaftlich, entwicklung
com_28	offen, gesellschaft
com_29	qualitativ, hochwertig
com_30	rechnung, tragen
com_31	rechtlich, rundfunk
com_32	regel, regeln
com_33	schnell, internet
com_34	schon, heute
com_35	schwarz, gelb
com_36	sexuell, orientierung
com_37	start, ups
com_38	stelle, stellen
com_39	strukturschwach, region
com_40	stunde, stunden
com_41	teil, teilen
com_42	treffen, triefen
com_43	verein, vereinen
com_44	vereint, nation
com_45	vgl, kapitel

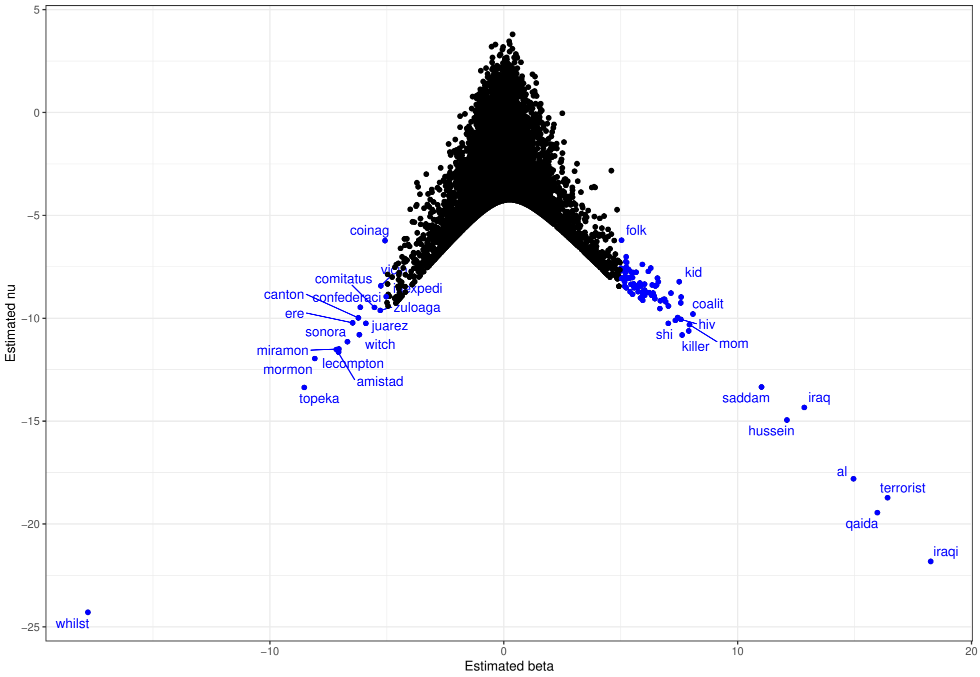


Figure 5: Word contributions from *Wordfish* (SOTU Corpus) (Diaf and Fritsche, 2022a)

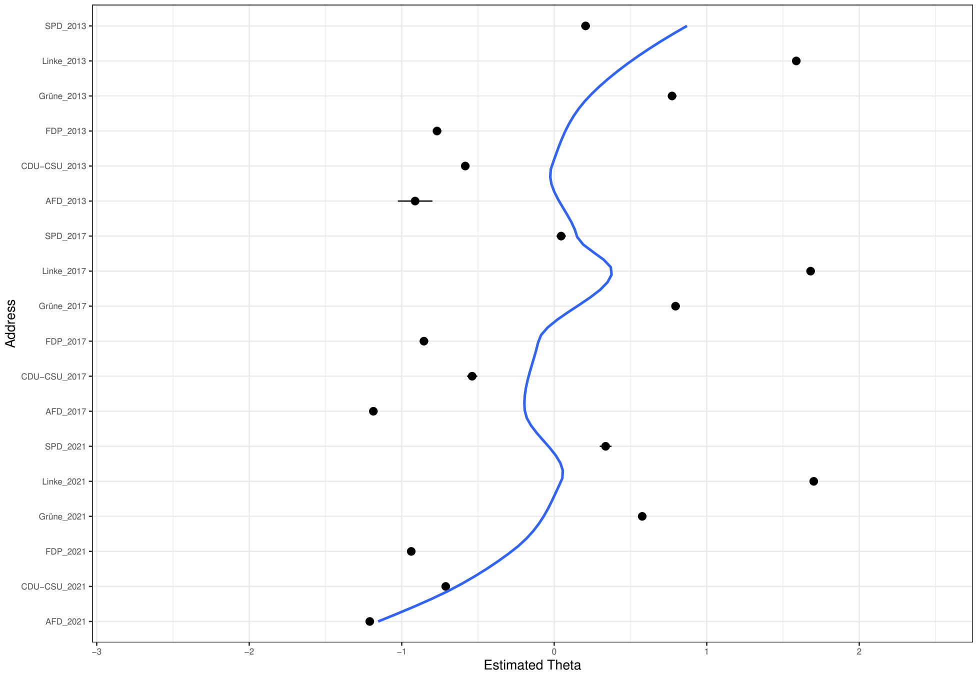


Figure 6: Learned *Wordfish* ideal points with 95% confidence intervals (German Manifesto Corpus). Blue line is the Loess curve.

References

Christopher A. Bail. 2016. Combining natural language processing and network analysis to examine how advocacy organizations stimulate conversation on social media. *Proceedings of the National Academy of Sciences*, 113(42):11823–11828.

Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

Aaron Clauset, M. E. J. Newman, and Cristopher Moore. 2004. Finding community structure in very large networks. *Phys. Rev. E*, 70:066111.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Sami Diaf and Ulrich Fritsche. 2022a. Topic scaling: A joint document scaling-topic model approach to learn time-specific topics. *Algorithms*, 15(11).

Sami Diaf and Ulrich Fritsche. 2022b. TopicShoal: Scaling partisanship using semantic search. In *Proceedings of the 18th Conference on Natural Language Processing (KONVENS 2022)*, pages 167–174, Potsdam, Germany.

Frederik Hjorth, Robert Klemmensen, Sara Hobolt, Martin Ejnar Hansen, and Peter Kurrild-Klitgaard. 2015. Computers, coders, and voters: Comparing automated methods for estimating party positions. *Research & Politics*, 2(2):2053168015580476.

William G. Jacoby. 2000. Loess:: a nonparametric, graphical tool for depicting relationships between variables. *Electoral Studies*, 19(4):577–613.

Carsten Jentsch, Eun Ryung Lee, and Enno Mammen. 2020. Time-dependent poisson reduced rank models for political text data analysis. *Computational Statistics & Data Analysis*, 142:106813.

Andrea Lancichinetti and Santo Fortunato. 2009. Community detection algorithms: A comparative analysis. *Phys. Rev. E*, 80:056117.

Benjamin E. Lauderdale and Alexander Herzog. 2016. Measuring political positions from legislative speech. *Political Analysis*, 24(3):374–394.

Pola Lehmann, Tobias Burst, Theres Matthieß, Sven Regel, Andrea Volkens, Bernhard Weßels, and Lisa Zehnter. 2022. The manifesto data collection. manifesto project (mrg/cmp/marpor). version 2022a.

James Lo, Sven-Oliver Proksch, and Jonathan B. Slapin. 2016. Ideological clarity in multiparty competition: A new measure and test using election manifestos. *British Journal of Political Science*, 46(3):591–610.

Will Lowe and Kenneth Benoit. 2013. Validating estimates of latent traits from textual data using human judgment as a benchmark. *Political Analysis*, 21(3):298–313.

Attila Mester, Andrei Pop, Bogdan-Eduard-Mădălin Mursa, Horea Greblă, Laura Dioşan, and Camelia Chira. 2021. Network analysis based on important node selection and community detection. *Mathematics*, 9(18).

Federico Nanni, Goran Glavas, Ines Rehbein, Simone Paolo Ponzetto, and Heiner Stuckenschmidt. 2019. Political text scaling meets computational semantics. *arXiv preprint arXiv:1904.06217*.

Martin Rosvall and Carl T. Bergstrom. 2008. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123.

Alix Rule, Jean-Philippe Cointet, and Peter S. Bearman. 2015. Lexical shifts, substantive changes, and continuity in state of the union discourse, 1790–2014. *Proceedings of the National Academy of Sciences*, 112(35):10837–10844.

Jonathan B. Slapin and Sven-Oliver Proksch. 2008. A scaling model for estimating time-series party positions from texts. *American Journal of Political Science*, 52(3):705–722.

Milan Straka, Jan Hajič, and Jana Straková. 2016. Udpipe: Trainable pipeline for processing conll-u files performing tokenization, morphological analysis, pos tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 4290–4297, Portorož, Slovenia. European Language Resources Association (ELRA).

Vincent A. Traag, Ludo Waltman, and Nees Jan van Eck. 2019. From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1).

Keyon Vafa, Suresh Naidu, and David Blei. 2020. Text-based ideal points. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5345–5357, Online. Association for Computational Linguistics.

Harrison C. White. 2008. *Identity and Control. How Social Formations Emerge (Second Edition)*. Princeton University Press, Princeton.

ADCluster: Adaptive Deep Clustering for Unsupervised Learning from Unlabeled Documents

Arezoo Hatefi
Umeå University
Sweden
arezooh@cs.umu.se

Xuan-Son Vu
Umeå University
Sweden
sonvx@cs.umu.se

Monowar Bhuyan
Umeå University
Sweden
monowar@cs.umu.se

Frank Drewes
Umeå University
Sweden
drewes@cs.umu.se

Abstract

We introduce ADCluster, a deep document clustering approach based on language models that is trained to adapt to the clustering task. This adaptability is achieved through an iterative process where K-Means clustering is applied to the dataset, followed by iteratively training a deep classifier with generated pseudo-labels – an approach referred to as *inner adaptation*. The model is also able to adapt to changes in the data as new documents are added to the document collection. The latter type of adaptation, *outer adaptation*, is obtained by resuming the inner adaptation when a new chunk of documents has arrived. We explore two outer adaptation strategies, namely accumulative adaptation (training is resumed on the accumulated set of all documents) and non-accumulative adaptation (training is resumed using only the new chunk of data). We show that ADCluster outperforms established document clustering techniques on medium and long-text documents by a large margin. Additionally, our approach outperforms well-established baseline methods under both the accumulative and non-accumulative outer adaptation scenarios.

1 Introduction

Document clustering is the task of arranging large volumes of unlabeled documents into clusters according to some notion of similarity. A particularly common goal is to discover the most common topics in a given collection of text documents and to assign each document to its corresponding cluster. Given the ever-growing number of documents available online and the fact that manually structuring them is impossible, there are countless applications of document clustering techniques.

General purpose clustering algorithms not specifically designed to work on text documents can be used for document clustering by creating vector representations of documents using deep neural networks and then clustering those vectors. One way of doing so is to use autoencoders (Ballard, 1987; Schmidhuber, 2015) applied to *term frequency – inverse document frequency* document representations (tf-idf, (Rajaraman and Ullman, 2011)). However, such representations neglect contextual information. Alternatively, one can use contextual representations obtained from pre-trained language models (LMs). Such approaches run a clustering algorithm such as K-Means over the output of the LM (Guan et al., 2022; Subakti et al., 2022; Grootendorst, 2022; Zhang et al., 2022; Eklund and Forsman, 2022). In another line of work, some studies proposed the simultaneous learning of document representations and clustering through a self-learning approach. This involves computing an auxiliary target distribution using the output of the model and minimizing the loss between these distributions (Huang et al., 2020; Xie et al., 2016; Hadifar et al., 2019). A problem with this approach is the risk of self-confirmation bias, potentially leading to trivial solutions. Moreover, the majority of these proposals rely on autoencoders, with limited exploration of LMs. In this paper, we introduce ADCluster, which uses K-Means as a teacher to

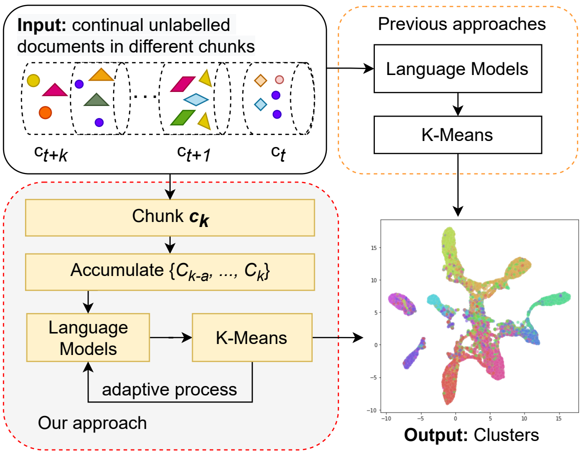


Figure 1: Overview of traditional approaches in comparison to ours in unsupervised text clustering tasks, where chunk data can be accumulated for the adaptive process.

train an LM-based classifier in an iterative manner to adapt it to the clustering task. Figure 1 shows the comparison between our approach and previous approaches (which use LMs) in the unsupervised clustering task. We hypothesize that the adaptation process is essential for any real-world application where there is no labeled training data.

In applications that rely on document clustering, the collection of documents is seldom static. For example, consider an online service using web crawlers to find new content of interest for them, or an online advertising service trying to discover appropriate web pages for ad placement (Hatefi et al., 2021). Given that new content is created every day, their document collections will steadily increase. With time, clustering will become unreliable because of subtle topic shifts or previously unknown terms such as Fridays for Future or King Charles III. Our method facilitates resuming the iterative adaptation of the model to the clustering task from its previous state when a new chunk of documents is to be incorporated.

Thus, we distinguish between inner and outer adaptation. Inner adaptation adjusts the LM to the clustering task at hand by an iterative training process during which the data is considered immutable. Outer adaptation adjusts the model over time to growing sets of documents by resuming the inner adaptation when a significant amount of new data becomes available, either by considering the entire dataset (*accumulative outer adaptation*) or using only the new data (*non-accumulative outer adaptation*). An obvious third possibility is to rebuild the model from scratch or use a scheduled combination of the three possibilities, depending on the practical conditions under which the model is used.

In this paper, we mainly focus on introducing the model and studying its performance under the accumulative and non-accumulative adaptation regimes. Future work will study the dynamic behavior arising when the model adapts to growing document collections as topics evolve.

Apart from introducing the clustering technique itself, and the algorithm used for training, we experiment with three different datasets, each of which we divide into five chunks in order to simulate growing collections of documents. The empirical results show the following:

1. Under each variant of the outer adaptation (training from scratch, accumulative, and non-accumulative adaptation), ADCluster outperforms the baselines.

2. In the absence of significant topic shifts, the three outer adaptation regimes usually result in comparable performance. Hence, one can choose between them as fits the application.

In addition to these main results, we conduct experiments to show that the method is insensitive to the type of language model used (our main experiments use BERT).

2 Related Work

Clustering is a much studied unsupervised problem in machine learning and data mining which is central to many data-driven applications. Many strategies for clustering arbitary sets of data points in an n-dimensional space have been studied. These include density-based, hierarchical, centroid- and partition-based clustering; see Xu and Tian (2015) for an overview. K-Means (MacQueen et al., 1967) and HDBSCAN (Campello et al., 2013) are two of the most popular traditional clustering algorithms.

The progress in deep learning that has been made during the last decade has made it natural to apply deep learning to clustering tasks (Zhou et al., 2022). An example of this is seen in DEC (Xie et al., 2016), which utilizes a stacked autoencoder to acquire document representations from tf-idf vectors. Subsequently, it improves these representations while learning clustering in a self-supervising manner. Hosseini and Varzaneh (2022) present a hybrid deep clustering method combining a stacked autoencoder and k-Means to organize Persian texts into clusters.

In recent years, large language models trained for language understanding and generation have achieved impressive results across a wide range of tasks. These LMs produce excellent general-purpose contextual representations that reflect topical information and can thus be used for clustering. Guan et al. (2022) generate document representations by pooling the outputs of ELMo (Peters et al., 2018) pre-trained LM and apply K-Means to these representations after normalizing them. Gupta et al. (2022) employ language models for unsupervised model interpretation and syntax induction through deep clustering of text representations. Huang et al. (2020) fine-tune the LM simultaneously with masked language modeling and clustering losses.

To our knowledge, no existing research explores deep clustering with LMs for dynamic scenarios involving a growing set of documents. Our method provides a simple yet effective approach to improve cluster assignments by training the LM in an adaptive manner to provide clustering-friendly representations that, over time, can be adapted to a growing set of documents.

3 Methodology

We first describe how the *inner* adaptation of the proposed model ADCluster works. Its pseudocode is given in Algorithm 1. It uses a conventional K-Means algorithm and a Deep Neural Network (DNN) classifier. The classifier is adapted iteratively in order to improve the clusterability of the embedding vectors. This is the inner adaptation. The classifier consists of a LM-based text encoder (a pre-trained LM with a mean pooling layer over its last layer) denoted by f_θ (where θ is the set of parameters) followed by a Multi-Layer Perceptron (MLP) head denoted by W that maps document representations to cluster assignments. Suppose we have an unlabeled dataset $D = \{d_n\}_{n=1}^{N}$ of N documents. At the beginning of each training epoch, we map each document d_n to its contextual representation $f_\theta(d_n)$. So, $E = \{f_\theta(d_n)\}_{n=1}^{N}$ is the set of document contextual representations. Often, it is beneficial to reduce the dimensionality of these representations using a dimension reduction method such as PCA (Pearson, 1901) or UMAP (McInnes et al., 2020), resulting in a set E' of vectors of fewer dimensions. Next, we use K-Means (based on cosine similarity rather than squared Euclidean distance) to cluster E' into K distinct clusters. We use these cluster assignments $\{p_n\}_{n=1}^{N}$ as *pseudo-labels* to train the classifier. For this, the MLP W and the encoder f_θ are jointly trained to minimize the cross entropy loss

$$\frac{\sum_{n=1}^{b} - \log \frac{\exp\left(y_{n,p_n}\right)}{\sum_{k=1}^{K} \exp\left(y_{n,k}\right)}}{b} \tag{1}$$

where y_n is the output of the classifier for document d_n and b is the mini-batch size. This cost function is minimized using AdamW (Loshchilov and Hutter, 2019) and backpropagation to compute the gradients. With the goal of preventing the classifier from overfitting to the current pseudo-labels, we employ only a subset of the data in every training epoch and restrict the number of iterations (i.e., *EpochSize* in Algorithm 1).

Algorithm 1: ADCluster (inner adaptation)

```
Input  : $D$: the set of unlabeled documents
         $f_\theta$: LM-based encoder of DNN classifier
         $W$: MLP head of DNN classifier
         $MaxIter$: the max training iterations
         $EpochSize$: iterations per training epoch
         $b$: the mini-batch size
         $\eta, \gamma$: the training learning rates
         $DR$: the dimension reduction method
         $\tau$: a threshold for the minimum
  percentage of changing assignments within two
  consecutive epochs (convergence threshold)
Output : $(\theta^*, W^*)$: The optimal weights
         $C$: final cluster assignments for $D$

$MaxEpoch \leftarrow MaxIter/EpochSize$;
for $epoch = 1$ to $MaxEpoch$ do
    $E \leftarrow$ encode $D$ with $f_\theta$;
    $E' \leftarrow DR(E)$        ▷ Apply DR with condition
    $P \leftarrow$ run K-means on $E'$ using cosine similarity;
    $X \leftarrow$ choose $b * EpochSize$ documents from
      pseudo-labeled set $P$ with a uniform sampler;
    $W \leftarrow$ initialize $W$ with Xavier initialization;
    for $iter = 1$ to $EpochSize$ do
        $B_{iter} \leftarrow$ choose a mini-batch from $X$;
        $Y_{iter} \leftarrow W(f_\theta(B_{iter}))$;
        $\hat{Y}_{\text{K-means}} \leftarrow P(B_{iter})$;
        $l \leftarrow$ cross-entropy-loss ($Y_{iter}$, $\hat{Y}_{\text{K-means}}$);
        $\theta \leftarrow \theta - \eta * l(\theta)$        ▷ Update $\theta$
        $W \leftarrow W - \gamma * l(W)$        ▷ Update $W$
    end
    $C_{curr} \leftarrow W_{predict}(f_\theta(D))$        ▷ predict cluster
      assignments for $D$ with DNN classifier
    $t \leftarrow$ compute ($C_{curr}$, $C_{prev}$)        ▷ Compute the
      percentage of changing cluster assignments
      compared to previous epoch;
    if $t < \tau$ then
        stop the iterative process
    end
    $C_{prev} \leftarrow C_{curr}$
end
return $\theta^*$,$W^*$,$C$;
```

It is worth mentioning that there is no correspondence between two consecutive cluster assignments. Hence, the final classification layer learned for an assignment becomes irrelevant for the following one and thus needs to be re-initialized from scratch at each epoch. We found that re-initializing the entire MLP head of the classifier rather than the final classifier layer is also beneficial for reducing the risk of overfitting. Since the MLP is a shallow network (having only one hidden layer), it can be trained sufficiently in one epoch.

In addition, we predict cluster assignments for all documents at the end of each epoch using the classifier and stop our procedure when the change in assignments is less than a threshold τ, i.e., the algorithm terminates when the number of documents for which the cluster assignment changes falls below τ.

Table 1: Datasets and statistics. *Silhouette Coefficient* refers to the Silhouette score of Rousseeuw (1987) which measures how similar a document is to its own cluster compared to other clusters, the best and worst values being 1 and -1, respectively. We compute the mean Silhouette Coefficient of all samples of the datasets using their true labels. As our LM for creating document representations, we use a BERT language model.

Dataset	Yahoo!5	Ag News	Fake News
#-Documents	38 812	40 000	480
Avg # sents	25.12	1.45	6.05
Avg # word (in doc)	578.26	36.09	141.20
Avg Silhouette Coefficient	0.01234	0.03736	0.04356

Overall, ADCluster alternates between clustering document representations to produce pseudo-labels and updating the parameters of the classifier by predicting these pseudo-labels using Eq. (1). This iterative adaptation of the encoder teaches the LM to generate more clustering-friendly representations. This distinguishes ADCluster from conventional methods, resulting in an improved K-Means clustering in subsequent epochs. The final clusters are obtained using the adapted classifier to predict cluster assignments.

If K-Means assigns almost all documents to a few large clusters, θ will only discriminate between them. A trivial parameterization occurs when all clusters except one are singletons, and therefore the classifier predicts the same output for all inputs (Caron et al., 2018). To overcome this problem, we train the classifier on uniformly sampled documents from the pseudo-labeled classes. The result is the same as weighting the contribution of a document to the loss function by the inverse of the size of the cluster to which it belongs.

Let us now briefly explain the *outer* adaptation of ADCluster. Imagine a data stream where new data arrives sequentially in chunks C_t, where t denotes the time step. In the *accumulative* scenario, we resume the inner adaptation of ADCluster at time t using $C_0 \cup \cdots \cup C_t$ as training data when a new chunk C_t arrives. In contrast, the *non-accumulative* approach resumes inner adaptation solely with the latest chunk C_t.

4 Experiments

4.1 Datasets

We employ the following three datasets whose statistics are summarized in Table 1:

Yahoo!5 is a subset of Yahoo! Answers (Zhang et al., 2015). The dataset comprises 10 classes, each document consisting of a question, a title, and the best answer to the question. We obtain the text to be clustered by concatenating these parts. To obtain a long-text dataset we only choose samples of over 500 tokens. The resulting dataset includes 38 812 documents.

Ag News (Zhang et al., 2015) consists of 4 classes: World, Sports, Business, and Sci/Tech news. The number of training and testing samples for each class is 30 000 and 1 900, respectively. We choose 40 000 documuments at random from the training set. To have a very short-text dataset, we only consider the news text and ignore the titles.

Fake News (Pérez-Rosas et al., 2018) comprises 480 medium-length news articles belonging to six different domains. While half of the articles are real and the other half are fake news, we do not make use of this distinction but use only the six topics of the dataset as labels.

Following the approach of prior studies (Huang et al., 2020; Xie et al., 2016; Hadifar et al., 2019), we form unlabelled documents by removing all labels for the training set, using the labels only to evaluate unsupervised performance.

4.2 Baselines

We use the following baselines for comparisons:

Traditional clustering algorithms We compare our model with K-Means and HDBSCAN. For HDBSCAN, we use the soft (or fuzzy) implementation[1] of the algorithm that predicts probability vectors for all dataset samples; no samples are considered noise. These vectors show the membership probability for each cluster, so we assign the sample to the cluster for which the highest probability has been determined. Instead of using pure BERT vectors, we apply normalization on them prior to performing dimension reduction and clustering. Before running HDBSCAN on the datasets, we perform dimension reduction using UMAP[2]. For each dataset, we test several values

[1] https://hdbscan.readthedocs.io/en/latest/soft_clustering.html

[2] https://umap-learn.readthedocs.io/en/latest/

for parameters of HDBSCAN and UMAP and report the highest accuracy we get. On Yahoo! Answers, we perform PCA dimension reduction ($n\_components = 0.8$; preserving at least 80% of variance) before K-Means.

DEC-tfidf we compare our model with that of Xie et al. (2016), using the available PyTorch implementation from `https://github.com/vlukiyanov/pt-dec`. We slightly adjust the parameters reported in the paper to our datasets and present the highest value obtained.

DEC-BERT To have a more fair comparison between ADCluster and DEC (Xie et al., 2016), we replace the stacked autoencoder part of DEC with a BERT language model followed by a mean pooling layer to encode documents and train it with the same objective function as in DEC.

UFT We compare our model with the model presented in Huang et al. (2020). We refer to this baseline as UFT. We obtained the source code from the authors of the paper and applied it to our datasets.

ADCluster-noIter is a non-iterative version of ADCluster. We run K-Means only once using contextual representations of documents from BERT and train the neural classifier with the generated pseudo-labels for some iterations.

Centroid-ADCluster Since in ADCluster there is no correspondence between two consecutive cluster assignments, the final classification layer learned for an assignment becomes irrelevant for the following one and thus needs to be reinitialized from scratch at each epoch. We do this to prevent the model from overfitting to the noisy pseudo-labels. For verification, we implemented another version of ADCluster in which we, instead of learning a classification layer predicting the cluster assignments, perform explicit comparisons between features and centroids.

4.3 Evaluation Metric

We adopt a standard unsupervised evaluation metric that is widely used in deep clustering studies to compare our proposed method to other algorithms. For all the algorithms, the number of clusters is set to the number of ground-truth categories of each dataset, and we evaluate the clustering performance using the unsupervised clustering accuracy (ACC):

$$ACC = \max_m \frac{\sum_{n=1}^{N} 1\{l_n = m(c_n)\}}{N}$$

where N is the total number of documents, l_n is the ground-truth label of document d_n, c_n is the cluster assignment that is predicted by the clustering algorithm for d_n, and m maps cluster assignments to labels, ranging over all possible one-to-one mappings. This metric seeks the best possible alignment between the ground-truth label and the cluster assignments generated by an unsupervised clustering algorithm. The Hungarian algorithm, presented in the work of Xu et al. (2003), offers a means to efficiently calculate the most effective mapping function within the context of a linear assignment problem.

4.4 Experimental Setup

We implemented ADCluster using the PyTorch framework, utilizing bert-base-uncased LM of Hugging Face[3]. Documents are truncated to their first 256 tokens. To generate document embeddings, we employ average pooling over the output of the language model. For label prediction, we employ a two-layer MLP with a single hidden layer. The hidden layer size is set to 128 for Yahoo!5 and Fake News and 768 for Ag News. The hyperbolic tangent function is used as the activation function for the MLP.

We set the mini-batch size to 4 and the learning rate of the LM and MLP head to 10^{-6} and 10^{-4} correspondingly. We also use a cosine scheduler for the learning rate of the LM. We train ADCluster for at most 10 000 iterations and reassign the clustering labels by applying K-Means on document representations every 200 iteration (which we call an *epoch*). The threshold for stopping training when cluster assignments do not significantly change anymore is set to 1% of the documents. The model is trained using the AdamW optimizer with α and β equal to 0.999. We use the first 200 iterations as warm-up steps for the LM. To initialize the centroids of K-Means we use the K-Means++ seeding strategy proposed by Arthur and Vassilvitskii (2007) and to initialize weights of MLP head in each epoch we use Xavier initialization (Glorot and Bengio, 2010). We train ADCluster-noIter and Centroid-ADCluster under the same settings. The only difference for Centroid-ADCluster is that the size of the hidden layer of the MLP head is 768 for all datasets and the weights of the last layer ($768 \cdot K$, where K is the number of classes in the dataset) are initialized with the centroids of the K-

[3] `https://huggingface.co/bert-base-uncased`

Means which are constant during training. For the other baselines, we test several sets of values for their hyperparameters and report the best results.

5 Results and Discussions

5.1 Overall Performance

Generally, ADCluster achieves better performances than most of the baseline methods across multiple datasets (see Table 2). Compared to traditional clustering algorithms, ADCluster outperforms K-Means from 1.84% (Ag News) up to 23.3% (Yahoo!5), indicating that the iterative learning process (inner adaptation) of our model is effective. We can also note that HDBSCAN achieves better performance than K-Means in most cases but outperforms ADCluster only in the case of Ag News. In Table 1, we see that Ag News consists of very short texts, its average number of sentences per document being 1.45 and the average number of words being 36.09. It does not seem to provide enough context for BERT to make distinctive representations, thus limiting the efficacy of our model on this particular dataset. However, in Section 5.5 we will see that by replacing BERT with more advanced LMs the performance of our model on this dataset improves. For Yahoo!5 and Fake News, HDBSCAN gains better performance than most of the other methods except ADCluster. In fact, for these datasets, ADCluster displays better performance than all baselines. This holds even in the case of Fake News, which consists of a very limited number of documents (i.e., 480 documents).

The comparison with DEC-based models yields the following observations. Firstly, ADCluster outperforms DEC-tfidf, which we attribute to its use of BERT contextual representations (whereas tf-idf representations only consider text as a bags of words and neglect their semantic relations). Secondly, even though DEC-BERT has similar access to the contextual information of the language model, its performance is still lower than that of our model. The same applies to the UFT baseline. The reason could be that these models are trained in a self-learning fashion and may thus suffer from self-confirmation. Our model avoids this by using K-Means as an external teacher for our neural classifier. It also uses a uniform sampling technique for batch creation, mitigating biases stemming from imbalanced clusters.

5.2 Dynamic Performance Analysis of ADCluster Across Varied Dataset Sizes

In this experiment, we examine the performance of ADCluster in comparison to baselines as the dataset size gradually increases. The outcomes of this experiment are presented in Table 3, illustrating the results as the document size expands from 10% to 100%. In general, ADCluster consistently maintains stable performance throughout these experiments and surpasses baseline models for all datasets, with the exception of the 10% case for Fake News.

5.3 Illustration of Learned Representations by ADCluster

In order to investigate how ADCluster develops clustering-friendly representations through internal adaptation, we visualize the evolution of clusters during the training process using the Yahoo!5 dataset. Figure 2 shows how ADCluster clusters the documents during different epochs with ground-truth classes represented by different colors. The figure clearly demonstrates that at the very beginning, the structure is random. Along with the adaptation process, documents are arranged into more distinct groups, which is signified by both color separation and spatial characteristics. This trend is further confirmed by the continuous enhancement in clustering performance observed in each successive epoch.

5.4 The Model Behavior on Data Streams

Notation. Hereafter, if not otherwise specified, we use *Ac* to abbreviate *Accumulation*. We randomly split each unlabelled data collection into 5 chunks and denote them by C_1 (1–20%), C_2 (21–40%), C_3 (41–60%), C_4 (61–80%), C_5 (81–100%).

We now analyze the outer adaptation behavior of ADCluster. In this experiment, we assume the number of the clusters to be constant over time, only receiving new samples. We compare our model with three baselines:

Word2vec+KM We generate document representations as the average of the Word2vec embeddings of all words in the document and use K-Means to cluster these representations.

BERT+KM We create document representations by taking the average of the output of the last BERT layer for non-pad tokens and use K-Means to cluster these representations.

ADCluster-scratch This baseline is the same

Table 2: Overall performances of ADCluster in comparison to baselines. ♥ indicates short-text datasets.

Method		Yahoo!5	Ag News♥	Fake News
Classic Clustering	Kmeans (BERT)	44.64	81.6	73.96
	HDBSCAN (BERT)	58.8	**83.68**	72.71
DEC (Xie et al., 2016)*	tf-idf	50.23	68.93	45.41
	BERT	46.43	78.32	75.83
UFT (Huang et al., 2020)*		46.94	65.46	66.67
ADCluster (ours)	Centroid-ADCluster	60.64	80.93	76.67
	ADCluster-Final	**67.94**	83.44	**77.50**

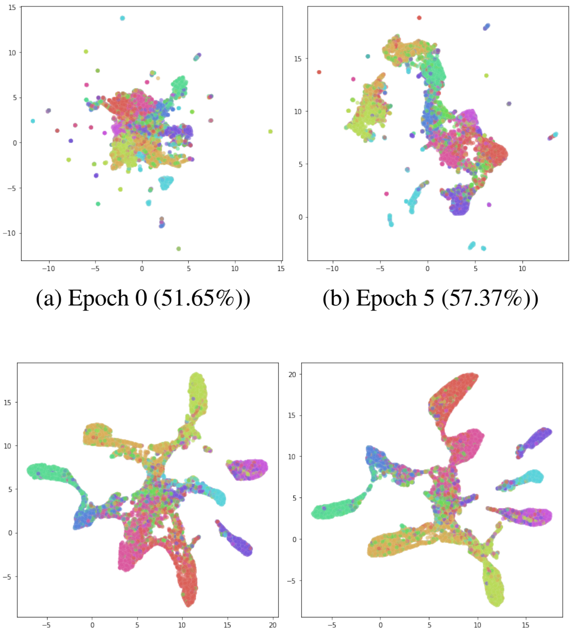
* The result is produced by us following the original paper

Table 3: Performance analysis of ADCluster across varied dataset sizes compared to baselines. Note that, because of the unsupervised setting, there is no expectation of monotonic increases in performance.

Dataset	Method	10%	50%	80%	100%
Ag News	K-Means	82.4	81.39	81.41	81.6
	DEC-BERT	79.3	78.22	78.4	78.32
	ADCluster	**84.08**	**82.56**	**84.3**	**83.44**
Yahoo!5	K-Means	53.23	53.5	59.95	52.17
	DEC-BERT	45.74	46.44	46.56	46.43
	ADCluster	**66.3**	**66.03**	**67.38**	**67.94**
Fake News	K-Means	64.58	77.08	77.34	73.96
	DEC-BERT	**68.75**	79.58	77.60	75.83
	ADCluster	64.58	**83.75**	**79.95**	**77.50**

as ADCluster except that instead of performing outer adaptation, we train the model from scratch (accumulatively on the whole dataset or non-accumulatively on the last chunk only, respectively). Thus, we remove the outer adaptation and the model only benefits from the inner adaptation. Tables 4–6 show the results of our experiments.

As our main take-aways from these experiments, we note that ADCluster outperforms the Word2vec+KM and BERT+KM baselines in all cases in both the *Ac* and *non-Ac* settings. The superior accuracy of ADCluster on chunk C_1 can



(a) Epoch 0 (51.65%))

(b) Epoch 5 (57.37%))

(c) Epoch 30 (67.53%))

(d) Epoch 50 (67.94%))

Figure 2: Illustration of clustered contextual representations according to ADCluster for Yahoo! Answer during inner adaptation. Colors indicate ground-truth classes. We have used UMAP to map 768-dimensional representations to a 2D feature space for illustration.

Table 4: Comparing the outer adaptation performance of ADCluster with baselines on Yahoo!5.

Method	Ac	C_1	C_2	C_3	C_4	C_5
Word2vec+KM	Yes	52.09	41.86	47.08	44.94	49.02
BERT+KM	Yes	46.28	53.84	53.67	55.24	53.70
ADCluster-scratch	Yes	**67.33**	66.44	64.06	64.51	62.06
ADCluster	Yes	**67.33**	**67.99**	**68.07**	**67.8**	**67.48**
Word2vec+KM	No	52.09	42.51	45.72	49.79	50.22
BERT+KM	No	46.28	57.02	52.00	54.86	55.04
ADCluster-scratch	No	**67.33**	67.11	65.19	61.79	65.50
ADCluster	No	**67.33**	**68.07**	**68.24**	**67.61**	**67.98**

Table 5: Comparing the outer adaptation performance of ADCluster with baselines on Ag News.

Method	Ac	C_1	C_2	C_3	C_4	C_5
Word2vec+KM	Yes	80.65	79.98	80.55	80.87	80.83
BERT+KM	Yes	81.66	81.42	81.50	81.51	81.52
ADCluster-scratch	Yes	**84.07**	84.56	**84.09**	**83.07**	81.76
ADCluster	Yes	**84.07**	**84.81**	82.56	83.05	**84.03**
Word2vec+KM	No	80.65	79.59	81.49	80.80	80.85
BERT+KM	No	81.66	81.43	81.20	81.82	81.05
ADCluster-scratch	No	**84.07**	83.74	81.95	**83.87**	82.51
ADCluster	No	**84.07**	**84.01**	**84.25**	83.6	**83.44**

Table 6: Comparing the outer adaptation performance of ADCluster with baselines on Fake News.

Method	Ac	C_1	C_2	C_3	C_4	C_5
Word2vec+KM	Yes	67.71	79.69	78.47	71.35	74.58
BERT+KM	Yes	57.29	77.60	77.08	77.34	77.29
ADCluster-scratch	Yes	**69.79**	82.81	**84.37**	79.69	79.58
ADCluster	Yes	**69.79**	**83.33**	83.68	**81.25**	**80.62**
Word2vec+KM	No	67.71	80.21	62.50	54.17	57.29
BERT+KM	No	57.29	77.08	58.33	53.12	51.04
ADCluster-scratch	No	**69.79**	82.29	67.71	57.29	59.37
ADCluster	No	**69.79**	**86.46**	**79.17**	**61.46**	**73.96**

be attributed to the inner adaptation which the baseline models lack. However, interestingly the outer adaptation results in superior performances in most cases on chunks C_2–C_5 even compared to ADCluster-scratch, which is remarkable and shows the effectiveness of outer adaptation.

5.5 Ablation study

In this ablation study, we design two settings to study the effectiveness of each ADCluster component. First, we replace the default BERT language model with recent models such as RoBERTa, SBERT, and BART. Second, we test various settings: (1) removing outer adaptation, (2) using a random sampler instead of a uniform sampler, and (3) Using UMAP for dimension reduction (instead of PCA for the Yahoo!5, and instead of not using dimension reduction for Ag News and Fake News). Figure 3 clearly shows that recent advanced language models yield better performance on all of the datasets. Table 7 summarizes the performance of ADCluster in the second setting. Across all experiments, the final model of ADCluster shows better performance than these variants.

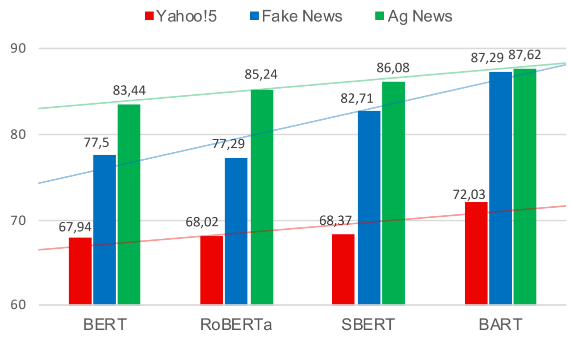


Figure 3: Ablation study w.r.t. different language models being used for the inner adaptation of ADCluster.

Table 7: Ablation study to evaluate the impact of different components of ADCluster to the final performance.

Ablation setting	Yahoo!5	Ag News	Fake News
Non iterative	53.89	82.88	73.96
UMAP	64.74	58.33	66.25
Random sampler	65.78	79.2	76.04

6 Conclusion and Future Work

We have introduced ADCluster, a neural document clustering model that iterates between a contextual language model and K-Means. K-Means is applied to contextualized document representations created by a BERT language model in order to obtain pseudo-labels. The weights of the language model are then iteratively adapted to improve the prediction of cluster assignments using discriminative loss. Not only does this *inner adaptation* result in superior clustering performance, it also enables us to resume training when the dataset grows (outer adaptation), as is often the case in real-world applications. Our empirical results show that for medium to long-text documents, ADCluster consistently outperforms conventional clustering models by a considerable margin with respect to the unsupervised accuracy measure.

Future work will have to study the inner and outer adaptation in more detail. For instance, one interesting direction could be a "soft adaptation", which continuously measures how much weight the outer adaptation shall place on earlier and later chunks. So far, we only presented two extreme cases, i.e., accumulation or non-accumulation.

Moreover, text data is often accompanied by additional modalities such as images, audio, and video. Such multimodal data has the potential to help the model understand the semantics of documents and assign them to the right cluster (Chen et al., 2021; Jiang et al., 2019). Multimodality can also open the door to new real-world downstream applications. Therefore, we are interested in extending our model to multimodal data clustering in the future.

References

David Arthur and Sergei Vassilvitskii. 2007. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, USA. Society for Industrial and Applied Mathematics.

Dana H. Ballard. 1987. Modular learning in neural networks. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, volume 1 of *AAAI'87*, page 279284. AAAI Press.

Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer Berlin Heidelberg.

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

Brian Chen, Andrew Rouditchenko, Kevin Duarte, Hilde Kuehne, Samuel Thomas, Angie Boggust, Rameswar Panda, Brian Kingsbury, Rogerio Feris, David Harwath, et al. 2021. Multimodal clustering networks for self-supervised learning from unlabeled videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8012–8021.

Anton Eklund and Mona Forsman. 2022. Topic modeling by clustering language model embeddings: Human validation on an industry dataset. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 635–643, Abu Dhabi, UAE. Association for Computational Linguistics.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *Computing Research Repository*, arXiv:2203.05794.

Renchu Guan, Hao Zhang, Yanchun Liang, Fausto Giunchiglia, Lan Huang, and Xiaoyue Feng. 2022. Deep feature-based text clustering and its explanation. *IEEE Transactions on Knowledge and Data Engineering*, 34(8):3669–3680.

Vikram Gupta, Haoyue Shi, Kevin Gimpel, and Mrinmaya Sachan. 2022. Deep clustering of text representations for supervision-free probing of syntax. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10720–10728.

Amir Hadifar, Lucas Sterckx, Thomas Demeester, and Chris Develder. 2019. A self-training approach for short text clustering. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 194–199, Florence, Italy. Association for Computational Linguistics.

Arezoo Hatefi, Xuan-Son Vu, Monowar Bhuyan, and Frank Drewes. 2021. Cformer: Semi-supervised text clustering based on pseudo labeling. In *Proceedings of the 30th ACM International Conference on Information &amp; Knowledge Management*, pages 3078–3082, Virtual Event, Queensland, Australia. Association for Computing Machinery.

Soodeh Hosseini and Zahra Asghari Varzaneh. 2022. Deep text clustering using stacked autoencoder. *Multimedia Tools Appl.*, 81(8):10861–10881.

Shaohan Huang, Furu Wei, Lei Cui, Xingxing Zhang, and Ming Zhou. 2020. Unsupervised fine-tuning for text clustering. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5530–5534, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yangbangyan Jiang, Qianqian Xu, Zhiyong Yang, Xiaochun Cao, and Qingming Huang. 2019. Dm2c: Deep mixed-modal clustering. In *Neural Information Processing Systems*.

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

James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297.

Leland McInnes, John Healy, and James Melville. 2020. Umap: Uniform manifold approximation and projection for dimension reduction. *Computing Research Repository*, arXiv:1802.03426. Version 3.

Karl Pearson. 1901. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Verónica Pérez-Rosas, Bennett Kleinberg, Alexandra Lefevre, and Rada Mihalcea. 2018. Automatic detection of fake news. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3391–3401. Association for Computational Linguistics.

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*, pages 2227–2237. Association for Computational Linguistics.

Anand Rajaraman and Jeffrey David Ullman. 2011. *Data Mining*, pages 1–17. Cambridge University Press.

Peter J. Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65.

Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117.

Alvin Subakti, Hendri Murfi, and Nora Hariadi. 2022. The performance of BERT as data representation of text clustering. *Journal of Big Data*.

Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, volume 48, pages 478–487, New York, NY, USA.

Dongkuan Xu and Yingjie Tian. 2015. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2:165–193.

Wei Xu, Xin Liu, and Yihong Gong. 2003. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273.

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.

Zihan Zhang, Meng Fang, Ling Chen, and Mohammad Reza Namazi Rad. 2022. Is neural topic modelling better than clustering? an empirical study on clustering with contextual embeddings for topics. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3886–3893, Seattle, United States. Association for Computational Linguistics.

Sheng Zhou, Hongjia Xu, Zhuonan Zheng, Jiawei Chen, Jiajun Bu, Jia Wu, Xin Wang, Wenwu Zhu, Martin Ester, et al. 2022. A comprehensive survey on deep clustering: Taxonomy, challenges, and future directions. *Computing Research Repository*, arXiv:2206.07579.

Efficient Black-Box Adversarial Attacks on Neural Text Detectors

Vitalii Fishchuk
University of Twente
Faculty of Electrical Engineering,
Mathematics and Computer Science
v.fishchuk@student.utwente.nl

Daniel Braun
University of Twente
Department of High-tech Business
and Entrepreneurship
d.braun@utwente.nl

Abstract

Neural text detectors are models trained to detect whether a given text was generated by a language model or written by a human. In this paper, we investigate three simple and resource-efficient strategies (parameter tweaking, prompt engineering, and character-level mutations) to alter texts generated by GPT-3.5 that are unsuspicious or unnoticeable for humans but cause misclassification by neural text detectors. The results show that especially parameter tweaking and character-level mutations are effective strategies.

1 Introduction

The widespread availability of neural text generation models, like ChatGPT, has caused an increased desire for neural text detectors, i.e. models that can detect whether a given text was AI-generated. The reliance of, e.g., educational institutions on such detectors has raised questions about their robustness in general and in specific with regard to adversarial attacks (Jawahar et al., 2020; Wolff and Wolff, 2022; Liang et al., 2023a,b). Such attacks exploit the fact that machine learning models by identifying patterns in the data rather than by understanding actual underlying concepts. Consequently, introducing small, human-unnoticeable perturbations can result in misclassification. (Goodfellow et al., 2014; Szegedy et al., 2013)

Adversarial attacks can be categorised into black-box and white-box attacks (Peng et al., 2023). In white-box attacks, the attacker has full access to the target model, including its parameters, architecture, and loss function (Ebrahimi et al., 2018; Gao et al., 2018). During black-box attacks, the adversary can only input queries and observe the outputs without any insights into internal processing (Gao et al., 2018). Furthermore, it can be distinguished between targeted and untargeted attacks, where targeted attacks aim at triggering misclassification towards a specific label, while untargeted aim to cause any misclassification (Rathore et al., 2021).

This paper investigates effective and resource-efficient universal attack strategies in a black-box scenario with minimal resources, based on text generated with GPT 3.5 and three neural text detectors: the widely used open source GPT-2 Output detector model[1], the OpenAI text classifier[2], and the commercial Turnitin AI detector[3], which is used by many educational institutions. The results show that character-level mutations, tweaking the parameters of the generative model, as well as prompt engineering, are efficient and effective strategies, showing that currently available neural text detectors can not reliably detect texts generated by state-of-the-art large language models (LLMs).

2 Related work

Most of the existing literature about adversarial attacks focuses on image detection. Textual input is less used due to its discrete nature and the difficulty in introducing human-imperceptible perturbations, contrary to the image data, where a change in a few hundred pixels can go unnoticed (Jin et al., 2019; Peng et al., 2023). Examples of adversarial attacks on general text classification models include the work by Ebrahimi et al. (2018) and Gao et al. (2018). More recent work has started to specifically look into adversarial attacks on neural text detectors: Wolff and Wolff (2022) showed that introducing spelling mistakes and replacing characters with homoglyphs can significantly reduce the detection rate for GPT-2 texts. Liang et al. (2023a) showed that similar character-level mutation-based attacks are also successful for RoBERTa-based detection models. Liang et al. (2023c) not only showed that

[1] https://github.com/openai/gpt-2-output-dataset/tree/master/detector

[2] https://platform.openai.com/ai-text-classifier

[3] https://www.turnitin.com

existing detectors are vulnerable to simple rephrasing, but they also showed that they are biased towards flagging texts that have been (manually) written by non-native speakers as AI-generated.

Because currently available methods are vulnerable to adversarial attacks, multiple suggestions have been made to improve their robustness, e.g. by Liang et al. (2023b), Shen et al. (2023), Crothers et al. (2022), and Yoo et al. (2022). While watermarking techniques to identify AI-generated texts are also investigated, they are generally seen as vulnerable to adversarial attacks, especially to mutation and paraphrasing-based approaches (Jin et al., 2019; Kirchenbauer et al., 2023; Sadasivan et al., 2023).

3 Approaches

Based on the existing literature, we identified three promising and efficient approaches for adversarial attacks: parameter tweaking, prompt engineering and character-level mutations. All approaches were tested with the three neural text detectors mentioned in Section 1: GPT-2 output detector, OpenAI classifier, and Turnitin AI writing detector. The basis for all attacks were texts generated by the GPT-3.5-turbo model via the OpenAI API. The text samples were produced as 500-word essays, with topics taken from a list of 200 essay topics (Nova, 2019) and the prompt *"Write a five-hundred-word argumentative essay on the topic 'topic'."*. It was then evaluated how the detection rate changed between the original texts and their altered version. To ensure comparability of the results between different detectors, all scores were projected onto a scale from 0.0 (very likely not AI-generated) to 1.0 (very likely AI-generated). The GPT-2 Output detector returns a score between 0.0 and 1.0, that can be used directly. Turnitin returns a percentage between 0 and 100 indicating how much of the text was generated by AI. We divide the score by 100. The OpenAI classifier returns one of five labels ("very unlikely", "unlikely", "unclear", "possibly", "likely"). For each of the labels, OpenAI (2023a) provides a corresponding range of numerical thresholds, of which we take the mean score (0.05, 0.275, 0.675, 0.94, 0.99). The code for the evaluation was written with the assistance of GPT-4, followed by extensive testing of the code, as well as additional, manually implemented, features. The code and the data are available on GitHub[4].

Parameter	Min	Max	Default
Temperature	0.0	2.0	1.0
Top p	0.0	1.0	1.0
Frequency penalty	-2.0	2.0	0.0
Presence penalty	-2.0	2.0	0.0

Table 1: Investigated parameters

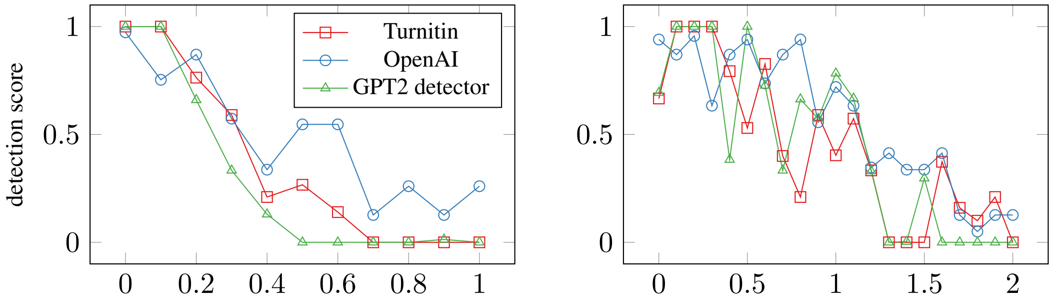
3.1 Parameter tweaking

First, we investigated the influence of GPT-3.5 generation parameters on the detection. Table 1 shows the parameters we focus on because they have the biggest impact on the produced texts according to OpenAI (2023b). Temperature and top p control randomness in the text. By increasing the temperature, the output becomes more random. However, for values beyond the default of 1.0, the length of the outputs started fluctuating strongly and the quality of the texts dropped. Consequently, we focused on the range between 0.0 and 1.0. Top p represents the percentage of tokens selected based on their probability mass. The frequency penalty controls the frequency of tokens appearing in the text, with higher values leading to more diverse verbatim. During the testing phase, it was found that increasing the frequency penalty beyond 1.0 degrades the quality of the texts rapidly. Additionally, as decreasing the value below 0.0 increases the repetitiveness, we focused on the range between 0.0 and 1.0. Finally, the presence penalty controls the model's likelihood of repeating tokens in the text. Higher values of presence penalty lead to the model producing more diverse texts. Following consideration similar to the frequency penalty, the negative values were discarded, and we focused on the range between 0.0 and 2.0. (OpenAI, 2023b) First, we investigated each parameter separately, changing it in steps of 0.1. Subsequently, we used the two parameters that had the biggest impact and investigated whether their interaction also influences the detection rate by performing a grid search in steps of 0.1 with these parameters.

3.2 Prompt engineering

The second approach explored the effect of prompt engineering on detection rates. Since Liang et al. (2023c) have shown that detection is vulnerable towards simple rephrasing, our hypothesis was that

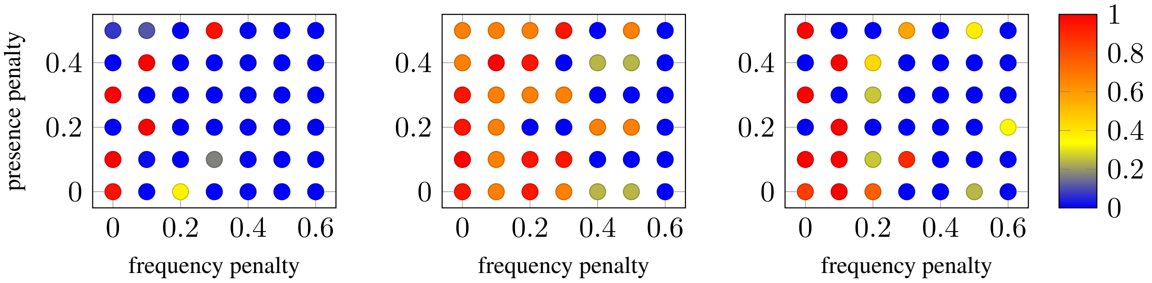
[4] https://github.com/Lolya-cloud/adversarial-attacks-on-neural-text-detectors



(a) frequency penalty

(b) presence penalty

Figure 1: Influence of the frequency and presence penalty on the detection score



(a) GPT-2 detector

(b) OpenAI classifier

(c) Turnitin

Figure 2: Influence of joined optimsation of fequency and presence penalties on the detection score

providing regeneration instructions as a separate prompt might lower the detection rate. This hypothesis was tested with the following four prompts, each of which was used to generate ten texts: (1) standard prompt as described in Section; (2) standard prompt followed by second query *"Regenerate the essay."*; (3) advanced prompt as shown in Appendix A.1; (4) standard prompt followed by advanced prompt as second query. Additionally, several prompts to increase the perplexity and burstiness of generated texts were tested, a strategy that is popular among users (see e.g. Alexander (2023)). Ten prompts were designed, and for each, ten texts were generated. The first five prompts use single-query architecture, while the others utilise a two-query concept. The following methods were tested with different prompts: (1) Explaining burstiness and perplexity, then asking to implement them (see Appendix A.2); (2) Explicitly asking to maximise either perplexity, burstiness or both (standard prompt followed by *“Maximize the burstiness / perplexity / burstiness and perplexity of the text.”*); (3) Explicitly asking to rewrite to avoid detection. (*“Rewrite the above essay in order to avoid AI detection.”*)

3.3 Character-level mutations

This approach aimed to test the robustness of the detectors against traditional adversarial attack vectors. Three character level mutations were taken as a basis: replacing either Latin lowercase “a” or “e” with the corresponding Cyrillic analogue; replacing Latin lowercase “L” with Latin uppercase “I”. Ten texts were generated with the standard prompt and then the mutations were applied.

4 Results

4.1 Parameter tweaking

The detection rate of all three detectors dropped with an increment in either frequency or presence penalty (see Figure 1). Starting from a frequency penalty of 0.3-0.4 and a presence penalty of 1.0-1.2, the detection rate fell under 50% with some fluctuations. An analysis of the generated texts showed that increasing either the frequency or presence penalty led to more diverse texts. A higher frequency penalty caused a wider vocabulary variety. For values above 0.6, the occurrence of punctuation mistakes and unclear wordings quickly increased, making the texts difficult to read. For values between 0.0 and 0.6, increases in value caused an incremental increase in text complexity while preserving quality and readability. A higher presence penalty primarily influenced the diversity of perspectives and text engagement. However, for values above 0.6, the coherence and logical progression rapidly decreased and texts became less subject-focused. Therefore, increasing either frequency or presence penalty from the default of 0.0 up to 0.6 can be seen as successful attack strategies that significantly lowers detection while maintaining text quality. Increasing the temperature and top p value above the default was already excluded from the experimental design, because of the strong negative effects on text quality. Lowering either value resulted in more deterministic outputs leading to higher detection rates, therefore, tweaking those parameters is not a successful attack strategy.

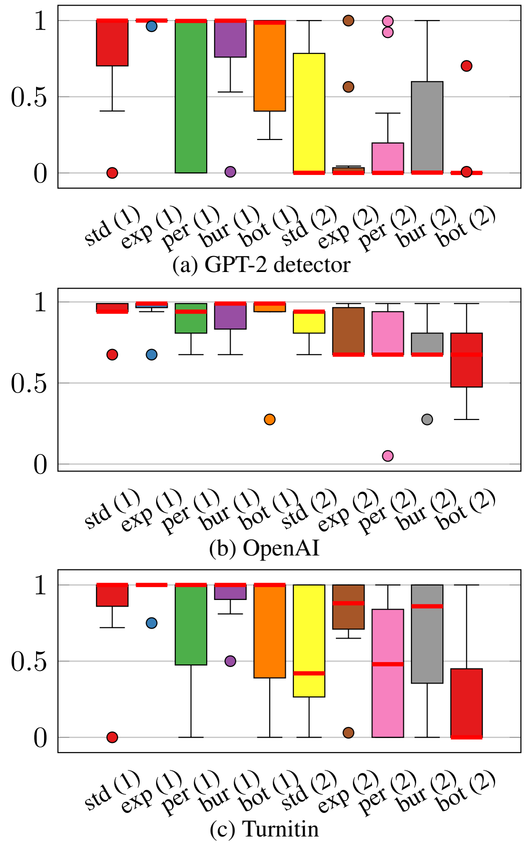
In a second step, the interaction between frequency and presence penalty was investigated. Figure 2 shows that the detection rates dropped for all three detectors with an increment in frequency and presence penalty. Notably, they started dropping for smaller values of presence and frequency penalty than they did for the individual parameters, thereby minimising the potential negative effects on text quality. The GPT-2 detector showed the worst performance in the comparison with very quickly declining detection scores.

4.2 Prompt engineering

The simple regeneration approach, whether using a second query or providing detailed instruction in the first query, did not have an impact on detection rates. Increasing the perplexity and burstiness of the texts through prompts, on the other hand, caused a drop in the detection rate across all three detectors, however only if applied in two separate prompts, as shown in Figure 3. Although the approach managed to decrease the detection rate across all three detectors, the score sank only for the GPT-2 detector below 0.5 (see Figure 3a). For the other two detectors, the score stayed above 0.5, meaning that the generated texts would still be detected as AI-generated or at least as undecidable.

4.3 Character-level mutations

The influence of the character-level mutations on the detection scores is shown in Table 2. For the GPT-2 detector, all three replacements lead to a detection score of 0. For Open AI, all attacks lowered the detection score, although not as much as for the GPT-2 detector. Turnitin detected the replacement of Latin characters with Cyrillic characters and flagged the attack. The substitution of *l*s with capital *i*s, however, remained undetected and significantly lowered the detection score, therefore presenting a successful attack strategy.



(a) GPT-2 detector

(b) OpenAI

(c) Turnitin

Figure 3: Influence of perplexity and burstiness prompts on detection (number of prompts in brackets; std = standard prompt; exp = prompt to explain and increase burstiness & perplexity; per = increase perplexity; bur = increase burstiness; bot = increase both)

5 Conclusion

This study explored the effectiveness of resource-efficient adversarial attacks on neural text detectors, based on texts generated by GPT-3.5 and the three detectors GPT-2 output detector, OpenAI classifier, and Turnitin. Of the three investigated strategies, parameter tweaking and character-level mutations were successful for all three detectors. Prompt engineering was only successful for the GPT-2 output detector. All strategies are resource efficient and easy to implement, effectively showing that currently available detectors cannot reliably detect AI-generated texts and are vulnerable to adversarial attacks.

	GPT-2	OpenAI	Turnitin
Standard	0.67	0.77	0.75
Swap a lat.-cyr.	0	0.52	x
Swap e lat.-cyr.	0	0.48	x
Swap l - I	0	0.38	0.21

Table 2: Character-level mutation mean detection score

References

Chris Alexander. 2023. Asking chatgpt to put perplexity and burstiness in an essay appears to fool ai detectors. https://telblog.unic.ac.cy/teaching-chatgpt-perplexity-burstiness-appears-to-fool-ai-detectors/. Last accessed: 2023-07-11.

Evan Crothers, Nathalie Japkowicz, Herna Viktor, and Paula Branco. 2022. Adversarial robustness of neural-statistical features in detection of generative transformers. *Proceedings of the International Joint Conference on Neural Networks*, 2022-July.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. *Proceedings - 2018 IEEE Symposium on Security and Privacy Workshops, SPW 2018*, pages 50–56.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks Lakshmanan, V.S. 2020. Automatic detection of machine generated text: A critical survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2296–2309, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, pages 8018–8025.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models.

Gongbo Liang, Jesus Guerrero, and Izzat Alsmadi. 2023a. Mutation-based adversarial attacks on neural text detectors.

Gongbo Liang, Jesus Guerrero, Fengbo Zheng, and Izzat Alsmadi. 2023b. Enhancing neural text detector robustness with &mu;attacking and rr-training. *Electronics*, 12(8).

Weixin Liang, Mert Yuksekgonul, Yining Mao, Eric Wu, and James Zou. 2023c. GPT detectors are biased against non-native english writers. In *ICLR 2023 Workshop on Trustworthy and Reliable Large-Scale Machine Learning Models*.

A. Nova. 2019. Essay topics: 100+ best essay topics for your guidance. https://www.5staressays.com/blog/essay-writing-guide/essay-topics. Last accessed: 2023-07-11.

OpenAI. 2023a. Ai text classifier - openai api. https://platform.openai.com/ai-text-classifier. Last accessed: 2023-07-11.

OpenAI. 2023b. Api reference - openai api.

Hao Peng, Zhe Wang, Dandan Zhao, Yiming Wu, Jianming Han, Shixin Guo, Shouling Ji, and Ming Zhong. 2023. Efficient text-based evolution algorithm to hard-label adversarial attacks on text. *Journal of King Saud University - Computer and Information Sciences*, 35:101539.

Pradeep Rathore, Arghya Basak, Sri Harsha Nistala, and Venkataramana Runkana. 2021. Untargeted, targeted and universal adversarial attacks and defenses on time series. *Proceedings of the International Joint Conference on Neural Networks*.

Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. 2023. Can ai-generated text be reliably detected?

Lujia Shen, Xuhong Zhang, Shouling Ji, Yuwen Pu, Chunpeng Ge, Xing Yang, and Yanghe Feng. 2023. Textdefense: Adversarial text detection based on word importance entropy.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*.

Max Wolff and Stuart Wolff. 2022. Attacking neural text detectors.

Ki Yoon Yoo, Jangho Kim, Jiho Jang, and Nojun Kwak. 2022. Detection of word adversarial examples in text classification: Benchmark and baseline via robust density estimation. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 3656–3672.

A Appendix - Prompts

A.1 Advanced Prompt

"Write a five-hundred-word argumentative essay on the topic 'topic'. Include personal reflections, use a mix of long and short sentences, employ rhetorical questions to engage the reader, maintain a conversational tone in parts, and play around with the paragraph structure to create a dynamic and engaging piece of writing. Try to include factual and contextual information, use advanced concepts and vocabulary. Utilize a combination of complex and simple vocabulary. Try to mimic human writing as closely as you can. Avoid passive voice, as it tends to occur more often in AI-generated texts. Add a few examples from the real world illustrating your point."

A.2 Perplexity and Burstiness

"Write a five-hundred-word argumentative essay on the topic 'topic'. When it comes to writing content, two factors are crucial, perplexity and burstiness. Perplexity measures the complexity of the text. Separately, burstiness compares the variations of sentences. Humans tend to write with greater burstiness, for example, with some longer or more complex sentences alongside shorter ones. AI sentences tend to be more uniform. Therefore, when writing the following content, I need it to have a good amount of perplexity and burstiness.

Transformer-Based Analysis of Sentiment Towards German Political Parties on Twitter During the 2021 Election Year

Nils Constantin Hellwig
Media Informatics Group
University of Regensburg
Regensburg, Germany
nils-constantin.hellwig@student.ur.de

Markus Bink
Media Informatics Group
University of Regensburg
Regensburg, Germany
markus.bink@student.ur.de

Thomas Schmidt
Media Informatics Group
University of Regensburg
Regensburg, Germany
thomas.schmidt@ur.de

Jakob Fehle
Media Informatics Group
University of Regensburg
Regensburg, Germany
jakob.fehle@ur.de

Christian Wolff
Media Informatics Group
University of Regensburg
Regensburg, Germany
christian.wolff@ur.de

Abstract

Twitter has become an important platform for political discussions among both politicians and the public and was extensively used during the 2021 federal election in Germany. Previous research examined the sentiment of the major political actors during that election on Twitter, but it remains unclear how the German public responded to them on Twitter in terms of sentiment. We analyzed a corpus of 713,742 tweets mentioning the Twitter handle of 89 of the most important party and politician accounts. We annotated a subset of 2,000 of these tweets regarding their sentiment and used this and other annotated corpora to implement and evaluate sentiment analysis algorithms based on single-label classification (positive, negative and neutral). We achieved best results with the German BERT model *gbert-large* using a combination of our annotated corpus and a previously annotated corpus from the same context as training material. This model achieves an average accuracy of 81.8% in a 5x5 cross-validation setting. Applying sentiment analysis on the overall corpus revealed that the majority of the tweets expressed negative sentiments. We investigated sentiment developments per party and show that sentiment was driven by significant events such as the implementation of stricter COVID-19 regulations.

1 Introduction

In 2021, the 20th German federal election took place, with the reigning chancellor Angela Merkel not running again after 16 years in office. After the election, Angela Merkel's party, the Christian Democratic Union (CDU), was no longer part of the government and a coalition was formed consisting of the Social Democratic Party (SPD), the Green Party (BÜNDNIS 90/DIE GRÜNEN), and the Free Democratic Party (Liberals, FDP). According to the opinion polling institute *Infratest Dimap*, a strong change in the political mood in the form of voting intention could be observed among voters during the election year.[1] Due to ongoing restrictions in the wake of the pandemic, campaigning by the respective parties on social media platforms like Twitter[2] played a special role in this election. Twitter is one of the most popular social media platforms and a micro-blogging platform where users can send out short posts ("tweets") which can then be viewed by other users. Tweets are limited to 280 characters (as of January 2023) and may also contain images, videos, links or hashtags, i.e. keywords marked with a "#"-sign. It is possible to mention other users in tweets by using their Twitter handle (e.g. @OlafScholz for the current German chancellor's account).

Twitter has become a popular platform for all sorts of analysis in Natural Language Processing (NLP) and Computational Social Science (CSS) including sentiment analysis. Sentiment analysis, also known as opinion mining, is the computational method to predict the sentiment, attitude, or opinion of media, predominantly text (Liu, 2020) and has major application areas in the analysis of social media (Schmidt et al., 2020), online reviews (Fehle

[1] https://www.infratest-dimap.de/umfragen-analysen/bundesweit/sonntagsfrage

[2] As of July 2023, Twitter has been rebranded as X. However, we will use the name "Twitter" in this paper since the data was acquired before the rebranding and "Twitter" is still a common reference for the platform.

et al., 2023), healthcare NLP (Moßburger et al., 2020) or narrative texts (Schmidt and Burghardt, 2018). The method has also been used extensively in the political context on Twitter to quantify both public sentiment towards political parties and actors (Agarwal et al., 2018; Yaqub et al., 2020), predict election results (Ibrahim et al., 2015; Ramteke et al., 2016), and to describe and relate sentiment of parties with one another (Tumasjan et al., 2011; Caetano et al., 2018). Previous research analyzed the tweets of major political accounts during the 2021 federal election in Germany and identified, among other things, a tendency towards negativity by opposition parties and significant sentiment changes before and after election day (Schmidt et al., 2022).

We build on this research but shift the focus from the political actors to the general public. In this paper, we perform sentiment analysis to analyze how the most important German political parties and their politicians were perceived on Twitter during the 2021 federal election campaign year. We have built a corpus of 713,742 tweets that were posted throughout the election year 2021 and that mention a selection of 89 political party accounts and politicians from the major German parties using their Twitter handle. Our research questions are as follows:

- How does the sentiment of the tweets differ comparing the major parties and comparing opposition and government parties?
- How does the sentiment expressed in tweets change over the course of the election year?
- How does the sentiment of tweets from political parties differ compared to tweets from users mentioning accounts of those parties?

Our main contributions are as follows:

- Acquisition and preparation of a corpus consisting of 713,742 tweets mentioning (using @-sign) 89 Twitter accounts by the major political German parties.
- Annotation of sentiment for a sub-corpus of 2,000 tweets.
- A fine-tuned and optimized German BERT model using annotations as training material.
- The analysis of classification results on the entire corpus focused on the proposed research questions.

Although sentiment analysis of the tweets of political actors during the 2021 German Federal Election has been already explored (Schmidt et al., 2022), to the best of our knowledge, no prior work has investigated citizens' sentiment during this election.

2 Related Work

2.1 Methods for Sentiment Analysis

Previous sentiment analysis research on Twitter has employed diverse approaches, ranging from lexicon-based methods (Elbagir and Yang, 2019; Hutto and Gilbert, 2014) to machine learning approaches like support vector machines (Awwalu et al., 2019; Xia et al., 2021), word embeddings (Lilleberg et al., 2015; Joulin et al., 2017) or neural networks (Zhang et al., 2018; Minaee et al., 2021; Xia et al., 2021). However, transformer-based models such as BERT (Devlin et al., 2019) and ELECTRA (Clark et al., 2020) which are trained on huge amounts of unlabeled textual data are currently considered state-of-the-art in a variety of NLP tasks including sentiment analysis (Dang et al., 2020; Qiu et al., 2020; Schmidt et al., 2021a). BERT-based models are available for many languages, and there are versions that have been fine-tuned on specific domains or languages. For example, for the German language, *deepset*[3] published large models that are trained on over 160 GB of German texts (Chan et al., 2020). In the context of Twitter, there are also some BERT-based models such as *BERTweet* (Nguyen et al., 2020) and *TwHIN-BERT* (Zhang et al., 2023) that have been fine-tuned on English tweets. In the field of political sentiment analysis, transformer-based models usually outperform lexicon-based methods and traditional machine learning methods (Chintalapudi et al., 2021; Fehle et al., 2021; Schmidt et al., 2022). Thus, we will focus on this approach for the implementation of our sentiment analysis.

2.2 Sentiment Analysis in the Context of Twitter for Political Research

Analyzing the sentiment of politicians' or political party tweets has been shown to accurately reflect the political orientation of these politicians or political parties. Tumasjan et al. (2011) found that the party sentiment profiles corresponded to how similar their political views between parties were.

[3] https://www.deepset.ai/

Additionally, politicians from opposing parties expressed opposing sentiments. Moreover, tweets during the 2016 American presidential election of both users and political actors have been used to identify homophily, i.e. “the tendency for individuals to interact with similar others”(Fu et al., 2012; Caetano et al., 2018). More recently, Schmidt et al. (2022) showed that in the 2021 German Federal election, the sentiment expressed in tweets of major parties was largely negative. Additionally, governing parties expressed more positive sentiments compared to those in the opposition.

Tweets have also been used to localize public opinion towards political actors in elections (Agarwal et al., 2018; Yaqub et al., 2020). Using both, geospatial data and sentiment analysis of tweets, Agarwal et al. (2018) have shown how political actors were perceived across the globe in the context of the EU Referendum regarding whether the UK should leave or stay in the EU. Likewise, Yaqub et al. (2020) evaluated the similarity between the sentiment of location-based tweets and on-ground public opinion and show that it corroborates with the election result. Similarly, Chaudhry et al. (2021) analyzed Twitter sentiment before, during, and after the 2020 US election on a state level. They find that the sentiment corresponded to a large degree with the final election results. Ali et al. (2022) investigated the sentiment expressed in tweets about Joe Biden and Donald Trump in the lead up to and aftermath of the 2020 US presidential election. Their findings indicate that following the election outcome, there was an increase in positive sentiment towards the winner Joe Biden. Using the public citizen sentiment of tweets regarding political candidates has also been shown to be useful in predicting the outcome of elections (Ibrahim et al., 2015; Ramteke et al., 2016).

3 Methodology

3.1 Data Acquisition

In order to capture the sentiment towards political actors on Twitter, we collected tweets that used the Twitter handles (using the @-sign) of a selection of accounts of parties represented in the Bundestag. The seven parties in the Bundestag were taken into account (whereby these were the same parties before and after the election). For each party, the ten politicians’ accounts with the most followers were considered (as of January 2022). In addition, tweets were collected that mentioned the three official party accounts with the most followers of the seven parties. Since the parties CDU and CSU form a parliamentary group and the CSU represents the state of Bavaria only, these parties were considered as a single party in the following analysis. Thus, the party accounts of CDU and CSU were summarized to four accounts. In total, 89 accounts were included in the analysis, which are the same as those considered by Schmidt et al. (2022) to enable direct comparisons between the sentiment expressed by political actors and the sentiment of public citizens towards them in the discussion. A full list of the accounts can be found in tables 6 and 7 in the appendix.

For the collection of tweets, we used *Twint*[4], a Python library that allows downloading large amounts of tweets. For each account, tweets were collected for two random days in each month of 2021. For one day, all tweets that mentioned the account with an @-sign were scraped. We have chosen these selection criteria due to resource and API limitations we would encounter when working with all tweets mentioning the 89 accounts for this year ($\sim$ 11 million tweets). We argue that the acquired corpus is still appropriate in size and representative in the context of our research goals.

Subsequently, tweets that did not have a German language code were filtered, as well as tweets in which the account under consideration mentioned itself. After filtering, the final corpus consists of 707,241 tweets and over 22 million tokens in total (see table 1 for general statistics of the corpus). The accounts of the parties SPD and CDU/CSU, which formed the government until the federal election, were mentioned in far more tweets and the party DIE LINKE the least compared to the other parties.

3.2 Data Annotation

We annotated a subset of randomly selected 2,000 tweets in order to train a machine learning model. The proportion of tweets related to a party in the annotated subset corresponded to the proportion in the entire corpus. The tweets were annotated independently of each other by five native speakers who were students or research assistants. Annotators received an annotation manual and a guided instruction session. Each tweet of the annotation subset should be assigned to one of the following sentiment labels by the annotators:

[4] https://github.com/twintproject/twint, https://github.com/kevctae/twint

Mentioned Party	Political Orientation	Pre-Election	Post Election	# Tweets	%	# Tokens	avg. Tweet Length
SPD	center left	government	government	228,415	32.3	7,153,549	31.32
CDU/CSU	center right	government	opposition	227,683	32.2	7,097,145	31.17
DIE GRÜNEN	left, ecological	opposition	government	73,261	10.4	2,408,946	32.88
FDP	liberal	opposition	government	79,815	11.3	2,607,610	32.67
AfD	far right	opposition	opposition	57,572	8.1	1,636,144	28.42
DIE LINKE	far left	opposition	opposition	40,495	5.7	1,340,331	33.10
Total	-	-	-	707,241	100	22,243,725	31.45

Table 1: General corpus statistics.

1. **positive**: Tweet has a predominantly positive connotation.
2. **negative**: Tweet has a predominantly negative connotation.
3. **neutral**: Tweet has a neutral sentiment tone.
4. **mixed**: Tweet contains positive and negative elements, with no predominant tendency towards positive/negative connotation.

Examples of annotations are shown in table 4 in the appendix. We acquired three annotations per tweet. Fleiss‘ κ and Krippendorff's α were calculated to measure the inter-rater agreement. Both Fleiss‘ κ and Krippendorff's α are 0.61; percentage-wise agreement is on average 66%. These values point towards substantial agreement according to Landis and Koch (1977).

Annotation	# Tweets	Proportion
positive	**120**	6,00%
negative	**976**	48,80%
neutral	**777**	38,85%
mixed	87	4,35%
no majority	40	2,00%

Table 2: Distribution of the sentiment classes of the annotated subset.

We assigned each tweet the majority annotation class and removed all tweets with no majority or mixed as majority annotation class since we perform sentiment analysis on a three class setting (neutral, positive, negative). The annotated corpus consists of 1,873 tweets after this filtering. The distribution of the majority labels for the annotated tweets is shown in table 2. The majority of tweets were annotated as negative (48.8%) while only few tweets are annotated as positive (6%).

3.3 Sentiment Analysis Model Training

Since large language models such as BERT are considered state-of-the-art in text classification, we decided to use *gbert-large* by *deepset*, a pre-trained model based on the BERT architecture (Chan et al., 2020) and one of the largest German language transformer-based models, as the base model. It also proved to be the best classification model in a similar setting (Schmidt et al., 2022). The model was loaded and implemented via *Hugging Face's*[5] model hub and fine-tuned for the downstream task of single-label classification on tweets with the classes: negative, positive and neutral. We used three different data sets for this fine-tuning process: (1) our 1,873 annotated tweets, (2) the 1,785 annotated tweets by Schmidt et al. (2022) which consists of tweets by politicians of the same election context and (3) the *GermEval 2017* dataset (Wojatzki et al., 2017). *GermEval 2017* consists of German sentiment-annotated posts from the field of customer feedback (Wojatzki et al., 2017) and is one of the most popular training corpus for sentiment analysis in German. We used the 26,209 annotated documents, referred to as the “main dataset” by Wojatzki et al. (2017). We evaluated a total of 4 different approaches with these three datasets in a 5x5 stratified cross-validation setting:

- **BERT-1:** Using 80% of dataset (1) for training and evaluating the model accordingly with 20% for all 5 cross-validation runs.
- **BERT-2:** As of BERT-1 + dataset (2) for training.
- **BERT-3:** As of BERT-1 + dataset (3) for training.

[5] https://huggingface.co/

- **BERT-4:** As of BERT-1 + dataset (2) and (3) for training.

All models were trained for five epochs, with a batch size of 16 for both training and evaluation. AdamW (Loshchilov and Hutter, 2019) was used as optimizer with a learning rate of 5e-6. This hyperparameter setting proved to achieve best results in our experiments. All models were trained on an NVIDIA GeForce GTX 1080 Ti GPU with 11 GB VRAM. The evaluation was solely carried out on the respective subset of our annotated dataset. For the implementation of the models and the evaluation we used *Pytorch* (Paszke et al., 2019), *Transformers* (Wolf et al., 2020) and *scikit-learn* (Pedregosa et al., 2011).

4 Results

4.1 Evaluation of the BERT Models

The results for the evaluation are shown in table 3. For all metrics, the best performance was achieved with BERT-2. The average accuracy is 81.8%, with precision and recall being higher for the negative and neutral classes than for the positive class. The worst accuracy was achieved with BERT-3, although BERT-1 and BERT-4 are only slightly better.

4.2 Analysis of Classification Results

All 707,241 tweets in the corpus were then classified using BERT-2. Thus, the final fine-tuned model was trained with 3,658 tweets: 606 (17%) positive, 1,512 (41%) negative and 1,540 (42%) neutral. We first present distribution and word frequency results and follow up with the analysis of time-based sentiment progressions. Please refer to table 5 in the appendix for election results to support the analysis and interpretation of the data.

4.2.1 General Analysis

As figure 1 shows, the majority of tweets were classified as negative (54.4%). Looking at the parties individually, we can see that for each party over 50% of the tweets were classified as negative, which is about the same as for the annotation subset. Tweets mentioning AFD accounts have the largest share of tweets classified as negative, while the FDP has the smallest share. However, for each party, over 50% of the tweets were classified as negative. Furthermore, it can be observed that the sentiment of the tweets that mentioned the parties that formed a government after the election (SPD, DIE GRÜNEN, FDP) have the lowest proportion of negative tweets in comparison to the parties that would become the opposition (CDU/CSU, AfD, DIE LINKE). This might be due to an overall more positive representation in the public after the election for the winning parties.

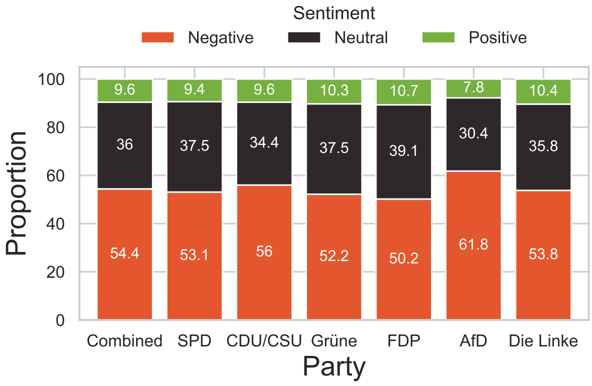


Figure 1: Distribution of the sentiment annotation for all parties.

For preliminary semantic analysis, we investigated the word frequencies of the three sentiment classes, looking for the most common positive or negative terms. In order to enhance the interpretability of the results, we removed stop words and all @-mentions from the tweets. Analyzing the word frequencies in negative tweets revealed frequent occurrences of terms such as "Corona", "Merkel" or "Impfung" (German for vaccination), showing the importance of COVID in the political discourse of that year. Terms such as "Danke" (thank you), "gut" (good) or "Herzlichen Glückwunsch" (congratulations) are most common among the positive tweets, indicating that post-election celebrations were the major source for positive tweets. Word clouds illustrating the word frequencies of all negative and positive tweets are presented in figures 4 and 5 in the appendix.

4.3 Diachronic Sentiment Analysis

We also carried out a diachronic sentiment analysis (similar to Schmidt et al., 2022). Tweets that were classified as positive were assigned +1, neutral tweets 0, and negative tweets -1. These values were then aggregated for tweets of each month and party for the election year 2021 and a mean sentiment score was calculated by averaging this value with the number of all tweets of that month and party (see figure 2). It is noticeable that the parties' curves are often in sync with each other and are

	BERT-1 $_{1}$	**BERT-2** $_{1+2}$	**BERT-3** $_{1+3}$	**BERT-4** $_{1+2+3}$
Accuracy	80.1	**81.8**	79.7	80.4
F1 Macro	74.3	**77.5**	73.9	75.2
F1 Micro	80.1	**81.8**	79.7	80.4
F1 Weighted	80.0	**81.7**	79.5	80.2
Precision $_{\text{positive}}$	71.0	**71.0**	69.8	69.3
Precision $_{\text{negative}}$	83.5	**84.8**	83.0	83.8
Precision $_{\text{neutral}}$	77.2	**79.8**	76.9	77.6
Recall $_{\text{positive}}$	55.0	**66.7**	55.8	60.0
Recall $_{\text{negative}}$	86.2	**86.6**	85.2	85.4
Recall $_{\text{neutral}}$	76.4	**78.1**	76.3	77.2

1 = Our Annotations, 2 = Annotations by Schmidt et al. (2022), 3 = GermEval 2017

Table 3: Results of the training of the different BERT models for the classification of sentiment.

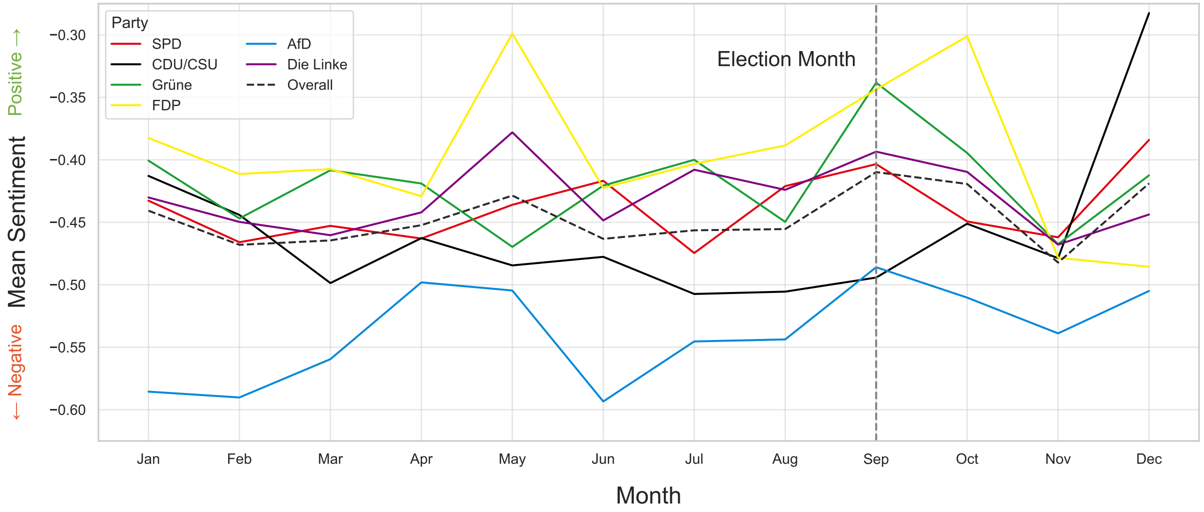


Figure 2: Mean sentiment of tweets mentioning the political accounts over the course of the election year.

constantly below -0.3 with only a few exceptions showing the dominant overall negativity in tweets that are mentioning political actors. It can be seen that the sentiment of the tweets deteriorated for all parties from January to February and October to November and improved from November to December. Tweets mentioning the AfD are the most negative for 11 months compared to the other parties, while for six months tweets mentioning the FDP are the most positive compared to the other parties.

To take a closer look at the period around election day, figure 3 shows the average sentiment of tweets mentioning the respective party for six weeks before and after the election day. Six weeks before the election day there are only small outliers but in general the sentiment remains approximately constant. Within the week starting on the election day, it is particularly noticeable that the sentiment of tweets mentioning Die Grünen was more positive compared to those mentioning Die Grünen in the previous week (about +0.4) and there is a clear outlier. The tweets mentioning the other two election winners (in terms of percentage gains) SPD and FDP were also more positive compared to the other three parties CDU, AfD and Die Linke, which recorded percentage losses in the election. Finally, within a week starting on 17 October, tweets that mentioned the SPD, Die Grünen and the FDP became more positive compared to the previous week, especially for the SPD (a major election winner) a clear change can be observed (about +0.3).

5 Discussion

In the following section, we discuss and interpret the overall results and highlight interesting findings. To discuss our third research question, we refer to

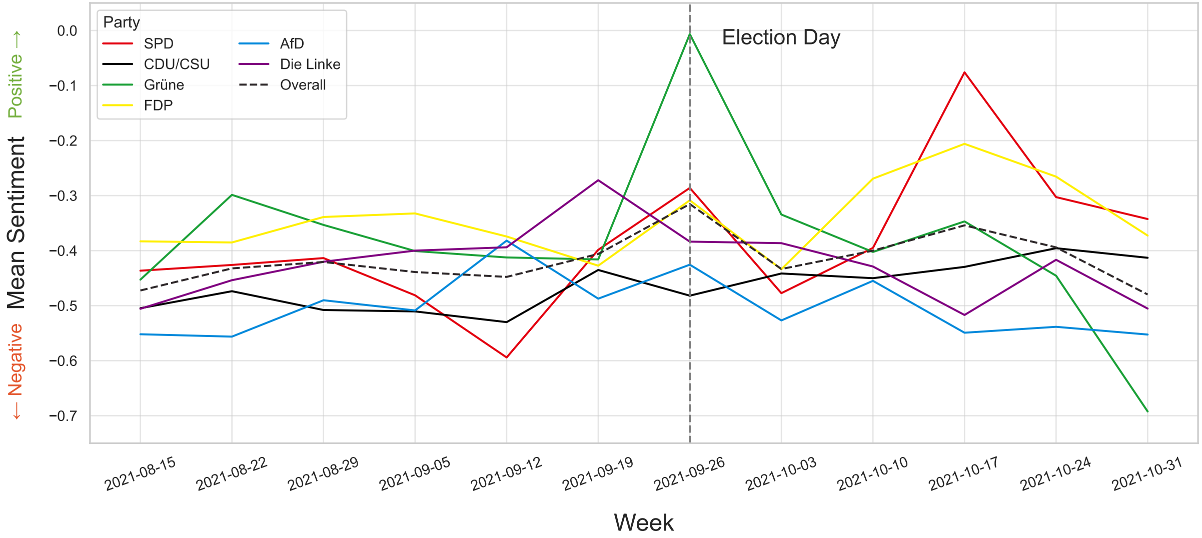


Figure 3: Mean sentiment of tweets mentioning parties over the course of 6 weeks before and after the election.

research by Schmidt et al. (2022) who did a similar study but analyzed the tweets of the 89 political actors themselves and not tweets mentioning them.

Considering the general corpus, it is noticeable that the parties in power until the election, CDU/CSU and SPD, were mentioned more often by tweets than the four opposition parties. Furthermore, it can be seen that the tweets are on average shorter than the tweets from the accounts of the political parties themselves. The tweets by the accounts of the political parties are 53.4 tokens long on average (Schmidt et al., 2022), whereas the tweets that mention the political parties, as shown in this paper, are only 31.45 tokens long. This may be because politicians, being in the public eye, are more cautious about the language and information they share. Conversely, public citizens may simply want to express their emotions towards others, and therefore, do not feel the same pressure to use more words and explain themselves in more detail.

We annotated a subset of 2,000 tweets from the corpus and achieved substantial agreement among the annotators. Sometimes, annotations showed disagreement, particularly in cases where tweets contained ironic and sarcastic language, or expressed mixed sentiments. This made it difficult for individual annotators to determine the overall sentiment of the tweet, resulting in varied interpretations. The annotated data set was then used to fine-tune a BERT model. The best of the evaluated models achieved an accuracy of about 81.8%. Methods of hyperparameter optimization and dealing with the class imbalance by assigning weights to labels for loss calculation during training showed no improvements. However, overall, the accuracies are in line with similar classification results in German (Chan et al., 2020).

Using the best model, we then classified the sentiment of the tweets of the entire corpus, with more than half of the tweets being classified as negative and less than 10% as positive. Compared to the tweets from the accounts of the political parties themselves (Schmidt et al., 2022), the sentiment is far less positive and more negative in average. This could possibly be attributed to politicians using positive and diplomatic language to gain support for their policies while avoiding offending anyone, whereas citizens tend to use negative language to express their frustration or dissatisfaction with political events or decisions. Regarding party-based classification results, we showed that tweets referring to the AFD were most often classified as negative compared to the other parties. Tweets about the election winner parties showed the most positive sentiment.

Subsequently, we analyzed how the sentiment in the tweets has evolved over the course of the election year. We identified several overall sentiment drops and peaks. The drop in sentiment observed in tweets from January to February could be explained by the ongoing discussions of state-level COVID-19 regulations during that period, as indicated by the corresponding term frequencies for these months. In these two months, terms such as “Lockdown”, “Pandemie” (pandemic) and “Corona” were frequently used in tweets. The drop in sentiment from October to November can probably also be explained by the fact that new COVID-19 restrictions were discussed after the summer, at which time similar terms were mentioned in the tweets. At the time of election day in September, the peak of DIE GRÜNEN is particularly noticeable and can be explained because they recorded the strongest percentage gain compared to the other parties. These findings are consistent with previous research (Ali et al., 2022) which has shown that there tends to be an increase in positive sentiment regarding the winning candidate around the time of the election and the announcement of the results. On the other hand, the CDU is the party where the tweets mentioning their accounts have the lowest sentiment in September and the week after the election compared to the other parties. Presumably, this can be explained since they lost the most votes in percentage compared to the last election. Finally, the increase in sentiment from November to December is likely due to the election of a new federal cabinet. This is supported by the most frequent words used in these months like "Glückwunsch", "Gratulation" (both congratulation) and "Erfolg" (success). Comparing the changes in sentiment over the year, there are differences and similarities comparing tweets from political party accounts (Schmidt et al., 2022) and tweets mentioning them, as studied in this paper. Our findings indicate that the parties do not have similar tops and lows and that the parties' courses of the sentiment are more asymmetric to each other. Nevertheless, the highs and lows in February, November and December are also recognizable and prove that major international and national events influence both in similar way politicians' tweets and tweets by the public about them. Our results also show that the sentiment of the AFD is more negative in most months compared to the other parties. But in comparison, the CDU is not the party whose tweets show the most positive sentiment in most months. Among the tweets we looked at, the CDU is the party with the most positive tweets only in December, most often it is the FDP.

6 Limitations and Future Work

Our work provides insights on how the political parties were perceived on Twitter in the election year 2021 and we contribute resources to the research area of sentiment analysis in German. However, there are limitations of our work that we intend to address in future work: Due to the high number of tweets mentioning party accounts, we decided not to collect tweets for all days within the election year and instead acquired tweets for two random days of each month in 2021. This certainly limits the representativeness of our corpus, since critical events or fluctuations in public sentiment may have been overlooked, like the 2021 European Floods, killing 196 in Germany[6] which had a strong impact in Germany during the election campaign. Furthermore, our corpus contains tweets that mention multiple accounts, which can dilute the sentiment targeted at the primary party or politician of interest. Another limitation is the accuracy of the trained model. While it is in line with similar studies and evaluation results (Chan et al., 2020), we plan to improve accuracies by annotating more tweets and exploring more methods of hyperparameter optimization. We want to address the performance also with more sophisticated methods to deal with class imbalance (Buda et al., 2018). Moreover, we will investigate the addition of more complex classes similar to emotion classification (Schmidt et al., 2021b; Dennerlein et al., 2023), as the annotators also reported that nuanced emotions occurred often. Furthermore, Twitter also offers multimedia content that we intend to explore via computer vision based sentiment- and emotion analysis (Schmidt et al., 2021c; Schmidt and Wolff, 2021; El-Keilany et al., 2022). Lastly, we also want to highlight that Twitter is not as popular in Germany as in other countries and thus represents a limited subsection of public social media sentiment. According to surveys, 10% of Germans use Twitter regularly[7] compared to 23% of U.S. adults.[8] In addition to that, we intend to improve upon the semantic exploration of our data via more sophisticated methods like topic modeling and named entity recognition. On the annotation side, we plan to investigate possibilities of more fine-grained annotation to gain a better understanding of the annotation theory on this material. Parts of our research and more information about this project are publicly available to support further research in this area.[9]

[6] Cf. https://en.wikipedia.org/wiki/2021_European_floods.

[7] https://de.statista.com/statistik/daten/studie/171006/umfrage/in-anspruch-genommene-angebote-aus-dem-internet/

[8] https://www.statista.com/statistics/232818/active-us-twitter-user-growth/

[9] https://github.com/NilsHellwig/Twitter_German_Federal_Election_Perception_2021

References

Amit Agarwal, Ritu Singh, and Durga Toshniwal. 2018. Geospatial sentiment analysis using twitter data for uk-eu referendum. *Journal of Information and Optimization Sciences*, 39(1):303–317.

Rao Hamza Ali, Gabriela Pinto, Evelyn Lawrie, and Erik J Linstead. 2022. A large-scale sentiment analysis of tweets pertaining to the 2020 us presidential election. *Journal of big Data*, 9(1):1–12.

Jamilu Awwalu, Azuraliza Abu Bakar, and Mohd Ridzwan Yaakub. 2019. Hybrid n-gram model using naïve bayes for classification of political sentiments on twitter. *Neural Computing and Applications*, 31(12):9207–9220.

Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. 2018. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259.

Josemar A Caetano, Hélder S Lima, Mateus F Santos, and Humberto T Marques-Neto. 2018. Using sentiment analysis to define twitter political users' classes and their homophily during the 2016 american presidential election. *Journal of internet services and applications*, 9(1):1–15.

Branden Chan, Stefan Schweter, and Timo Möller. 2020. German's next language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6788–6796, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hassan Nazeer Chaudhry, Yasir Javed, Farzana Kulsoom, Zahid Mehmood, Zafar Iqbal Khan, Umar Shoaib, and Sadaf Hussain Janjua. 2021. Sentiment analysis of before and after elections: Twitter data of us election 2020. *Electronics*, 10(17):2082–2108.

Nalini Chintalapudi, Gopi Battineni, and Francesco Amenta. 2021. Sentimental analysis of covid-19 tweets using deep learning models. *Infectious Disease Reports*, 13(2):329–339.

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*.

Nhan Cach Dang, María N Moreno-García, and Fernando De la Prieta. 2020. Sentiment analysis based on deep learning: A comparative study. *Electronics*, 9(3):483–512.

Katrin Dennerlein, Thomas Schmidt, and Christian Wolff. 2023. Computational emotion classification for genre corpora of German tragedies and comedies from 17th to early 19th century. *Digital Scholarship in the Humanities*, 38(4):1466–1481.

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.

Alina El-Keilany, Thomas Schmidt, and Christian Wolff. 2022. Distant Viewing of the Harry Potter Movies via Computer Vision. In *Proceedings of the 6th Digital Humanities in the Nordic and Baltic Countries Conference (DHNB 2022).*, pages 33–49, Uppsala, Sweden.

Shihab Elbagir and Jing Yang. 2019. Twitter sentiment analysis using natural language toolkit and vader sentiment. In *Proceedings of the international multiconference of engineers and computer scientists 2019*.

Jakob Fehle, Leonie Münster, Thomas Schmidt, and Christian Wolff. 2023. Aspect-based sentiment analysis as a multi-label classification task on the domain of german hotel reviews. In *Proceedings of the 19th Conference on Natural Language Processing (KONVENS 2023)*, Ingolstadt, Germany. KONVENS 2023 Organizers.

Jakob Fehle, Thomas Schmidt, and Christian Wolff. 2021. Lexicon-based sentiment analysis in German: Systematic evaluation of resources and preprocessing techniques. In *Proceedings of the 17th Conference on Natural Language Processing (KONVENS 2021)*, pages 86–103, Düsseldorf, Germany. KONVENS 2021 Organizers.

Feng Fu, Martin A Nowak, Nicholas A Christakis, and James H Fowler. 2012. The evolution of homophily. *Scientific reports*, 2(1):845.

Clayton Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the Eigth International AAAI conference on web and social media*, volume 8, pages 216–225.

Mochamad Ibrahim, Omar Abdillah, Alfan F Wicaksono, and Mirna Adriani. 2015. Buzzer detection and sentiment analysis for predicting presidential election results in a twitter nation. In *2015 IEEE international conference on data mining workshop (ICDMW)*, pages 1348–1353. IEEE.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

John Richard Landis and Gary Grove Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Joseph Lilleberg, Yun Zhu, and Yanqing Zhang. 2015. Support vector machines and word2vec for text classification with semantic features. In *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, pages 136–140. IEEE.

Bing Liu. 2020. *Sentiment analysis: Mining opinions, sentiments, and emotions*. Cambridge university press.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. Machine Learning Repository. arXiv:1711.05101. Version 3.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning–based text classification: A comprehensive review. *ACM Computing Surveys*, 54(3):1–40.

Luis Moßburger, Felix Wende, Kay Brinkmann, and Thomas Schmidt. 2020. Exploring Online Depression Forums via Text Mining: A Comparison of Reddit and a Curated Online Forum. In *Proceedings of the Fifth Social Media Mining for Health Applications Workshop & Shared Task*, pages 70–81, Barcelona, Spain (Online). Association for Computational Linguistics.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, Online. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 8026–8037, Red Hook, NY, USA. Curran Associates Inc.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Jyoti Ramteke, Samarth Shah, Darshan Godhia, and Aadil Shaikh. 2016. Election result prediction using twitter sentiment analysis. In *2016 international conference on inventive computation technologies (ICICT)*, volume 1, pages 1–5. IEEE.

Thomas Schmidt and Manuel Burghardt. 2018. An Evaluation of Lexicon-based Sentiment Analysis Techniques for the Plays of Gotthold Ephraim Lessing. In *Proceedings of the Second Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 139–149, Santa Fe, New Mexico. Association for Computational Linguistics.

Thomas Schmidt, Katrin Dennerlein, and Christian Wolff. 2021a. Emotion classification in German plays with transformer-based language models pretrained on historical and contemporary language. In *Proceedings of the 5th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 67–79, Punta Cana, Dominican Republic (online). Association for Computational Linguistics.

Thomas Schmidt, Katrin Dennerlein, and Christian Wolff. 2021b. Using Deep Learning for Emotion Analysis of 18th and 19th Century German Plays. In Manuel Burghardt, Lisa Dieckmann, Timo Steyer, Peer Trilcke, Niels-Oliver Walkowski, Joëlle Weis, and Ulrike Wuttke, editors, *Fabrikation von Erkenntnis: Experimente in den Digital Humanities. Teilband 1*. Melusina Press, Esch-sur-Alzette, Luxembourg.

Thomas Schmidt, Alina El-Keilany, Johannes Eger, and Sarah Kurek. 2021c. Exploring Computer Vision for Film Analysis: A Case Study for Five Canonical Movies. In *2nd International Conference of the European Association for Digital Humanities (EADH 2021)*, Krasnoyarsk, Russia.

Thomas Schmidt, Jakob Fehle, Maximilian Weissenbacher, Jonathan Richter, Philipp Gottschalk, and Christian Wolff. 2022. Sentiment analysis on Twitter for the major German parties during the 2021 German federal election. In *Proceedings of the 18th Conference on Natural Language Processing (KONVENS 2022)*, pages 74–87, Potsdam, Germany. KONVENS 2022 Organizers.

Thomas Schmidt, Florian Kaindl, and Christian Wolff. 2020. Distant reading of religious online communities: A case study for three religious forums on reddit. In *Proceedings of the Digital Humanities in the Nordic Countries 5th Conference (DHN 2020)*, pages 157–172, Riga, Latvia.

Thomas Schmidt and Christian Wolff. 2021. Exploring Multimodal Sentiment Analysis in Plays: A Case Study for a Theater Recording of Emilia Galotti. In *Proceedings of the Conference on Computational Humanities Research 2021 (CHR 2021)*, pages 392–404, Amsterdam, The Netherlands.

Andranik Tumasjan, Timm O Sprenger, Philipp G Sandner, and Isabell M Welpe. 2011. Election forecasts with twitter: How 140 characters reflect the political landscape. *Social science computer review*, 29(4):402–418.

Michael Wojatzki, Eugen Ruppert, Sarah Holschneider, Torsten Zesch, and Chris Biemann. 2017. GermEval 2017: Shared Task on Aspect-based Sentiment in Social Media Customer Feedback. In *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*, pages 1–12, Berlin, Germany.

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.

Ethan Xia, Han Yue, and Hongfu Liu. 2021. Tweet sentiment analysis of the 2020 u.s. presidential election. In *Companion Proceedings of the Web Conference 2021*, WWW '21, page 367–371, New York, NY, USA. Association for Computing Machinery.

Ussama Yaqub, Nitesh Sharma, Rachit Pabreja, Soon Ae Chun, Vijayalakshmi Atluri, and Jaideep Vaidya. 2020. Location-based sentiment analyses and visualization of twitter election data. *Digital Government: Research and Practice*, 1(2):1–19.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253.

Xinyang Zhang, Yury Malkov, Omar Florez, Serim Park, Brian McWilliams, Jiawei Han, and Ahmed El-Kishky. 2023. Twhin-bert: A socially-enriched pre-trained language model for multilingual tweet representations at twitter. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, page 5597–5607, New York, NY, USA. Association for Computing Machinery.

A Appendix

A.1 Annotation Examples

Annotation	Tweet	Author	Mentioned Account (Party)
positive	(Offenbar) Unpopular opinion: Ich mag @ArminLaschet als Persönlichkeit und kann ihn mir als Kanzler durchaus vorstellen.	@fredschorn	@ArminLaschet (CDU)
negative	@derspiegel Scholz versagt irgendwie, @Karl_Lauterbach, tun Sie etwas.	@1worldvs1virus	@Karl_Lauterbach (SPD)
neutral	@rRockxter @europeika @CDU Was hat denn die CDU mit dem Christentum zu tun?	@123JulianN321	@CDU (CDU)
mixed	@GrueneBundestag @BriHasselmann Grüne Verbots Partei.. ahnungslos Glücklich	@Paellamixta	@GrueneBundestag (Die Grünen)
no majority	@gb_1960 @SWagenknecht Die Gehirnwäsche hat gewirkt. Du hättest herrlich in die DDR gepasst.	@MaierJrg1	@SWagenknecht (Die Linke)

Table 4: Annotation Examples: For the first four tweets, the annotators were unanimous, the last example was annotated as neutral, positive and mixed (no majority).

A.2 Results of German Federal Election 2021

Party	Full Name	2021	2017	Change
SPD	Social Democratic Party of Germany	25.7 %	20.5 %	+ 5.2 %
CDU/CSU	Christian Democratic Union/ Christian Social Union (Bavaria)	24.1 %	32.9 %	- 8.8 %
Die Grünen	The Greens	14.8 %	8.9 %	+ 5.9 %
FDP	Free Democratic Party	11.5 %	10.7 %	+ 0.8 %
AfD	Alternative for Germany	10.3 %	12.6 %	- 2.3 %
Die Linke	The Left	4.9 %	9.2 %	- 4.3 %

Table 5: Election results of the 2021 federal election and changes compared to the previous election in 2017.

A.3 Twitter accounts for data acquisition

A.3.1 Parties

SPD	CDU	CSU	Die Grünen	FDP	AfD	Die Linke
@spdde Follower: 417k Tweets: 22,138	@CDU Follower: 378k Tweets: 37,100	@CSU Follower: 229k Tweets: 9,072	@Die_Gruenen Follower: 649k Tweets: 30,560	@fdp Follower: 414k Tweets: 27,981	@AfD Follower: 173k Tweets: 8,330	@dieLinke Follower: 350k Tweets: 14,135
@spdbt Follower: 217k Tweets: 9,809	@cducsubt Follower: 166k Tweets: 13,250		@GrueneBundestag Follower: 186k Tweets: 6,399	@fdpbt Follower: 39k Tweets: 8,194	@AfDimBundestag Follower: 68k Tweets: 4,713	@Linksfraktion Follower: 108k Tweets: 2,994
@jusos Follower: 77k Tweets: 1,847	@Junge_Union Follower: 79k Tweets: 931		@gruene_jugend Follower: 76k Tweets: 1,290	@fdp_nrw Follower: 28k Tweets: 884	@AfDBerlin Follower: 19k Tweets: 364	@dielinkeberlin Follower: 19k Tweets: 1,228

Table 6: The 3 main accounts with the most followers for each party (as of January 2022).

A.3.2 Politicians

SPD	CDU	CSU
@Karl_Lauterbach Follower: 770k Tweets: 132,526	@jensspahn Follower: 279k Tweets: 35,571	@Markus_Soeder Follower: 341k Tweets: 30,495
@HeikoMaas Follower: 660k Tweets: 6,431	@ArminLaschet Follower: 188k Tweets: 36,161	@DoroBaer Follower: 103k Tweets: 2,560
@OlafScholz Follower: 324k Tweets: 27,414	@_FriedrichMerz Follower: 179k Tweets: 23,651	@andreasscheuer Follower: 63k Tweets: 2,431
@KuehniKev Follower: 323k Tweets: 5,192	@JuliaKloeckner Follower: 74k Tweets: 3,357	@ManfredWeber Follower: 54k Tweets: 527
@larsklingbeil Follower: 116k Tweets: 5,669	@n_roettgen Follower: 68k Tweets: 4,645	@DerLenzMdB Follower: 10k Tweets: 236
@hubertus_heil Follower: 108k Tweets: 2,406	@PaulZiemiak Follower: 58k Tweets: 12,723	@hahnflo Follower: 9k Tweets: 2,900
@EskenSaskia Follower: 101k Tweets: 7,180	@groehe Follower: 49k Tweets: 79	@smuellermdb Follower: 9k Tweets: 239
@Ralf_Stegner Follower: 64.9k Tweets: 7,061	@HBraun Follower: 39k Tweets: 3,212	@DaniLudwigMdB Follower: 8k Tweets: 3,821
@KarambaDiaby Follower: 55.6k Tweets: 392	@rbrinkhaus Follower: 30k Tweets: 4,280	@ANiebler Follower: 6k Tweets: 25
@MiRo_SPD Follower: 39k Tweets: 350	@tj_tweets Follower: 17k Tweets: 396	@MarkusFerber Follower: 5k Tweets: 21

Die Grünen	FDP	AfD	Die Linke
@cem_oezdemir Follower: 290k Tweets: 9,942	@c_lindner Follower: 552k Tweets: 19,942	@Alice_Weidel Follower: 138k Tweets: 9,367	@SWagenknecht Follower: 518k Tweets: 7,177
@GoeringEckardt Follower: 202k Tweets: 5,227	@MaStrackZi Follower: 46k Tweets: 2,453	@Joerg_Meuthen Follower: 76k Tweets: 4,813	@GregorGysi Follower: 439k Tweets: 1,722
@JTrittin Follower: 115k Tweets: 1,782	@MarcoBuschmann Follower: 46k Tweets: 10,062	@Beatrix_vStorch Follower: 68k Tweets: 3,962	@katjakipping Follower: 130k Tweets: 1,072
@KonstantinNotz Follower: 85k Tweets: 2,144	@KonstantinKuhle Follower: 44k Tweets: 2,710	@gottfriedcurio Follower: 37k Tweets: 275	@DietmarBartsch Follower: 82k Tweets: 3,409
@RenateKuenast Follower: 77k Tweets: 2,026	@johannesvogel Follower: 38k Tweets: 2,121	@MalteKaufmann Follower: 36k Tweets: 5149	@anked Follower: 43k Tweets: 935
@Ricarda_Lang Follower: 65k Tweets: 3,546	@Wissing Follower: 32k Tweets: 2,805	@JoanaCotar Follower: 30k Tweets: 4,330	@b_riexinger Follower: 41k Tweets: 1,399
@KathaSchulze Follower: 37k Tweets: 4,609	@Lambsdorff Follower: 27k Tweets: 884	@Tino_Chrupalla Follower: 21k Tweets: 2,875	@jankortemdb Follower: 34k Tweets: 743
@BriHasselmann Follower: 37k Tweets: 1,795	@ria_schroeder Follower: 23k Tweets: 359	@StBrandner Follower: 23k Tweets: 11,914	@Janine_Wissler Follower: 37k Tweets: 1,046
@nouripour Follower: 29k Tweets: 505	@LindaTeuteberg Follower: 23k Tweets: 328	@GtzFrmming Follower: 17k Tweets: 984	@SevimDagdelen Follower: 35k Tweets: 172
@MiKellner Follower: 28k Tweets: 3,436	@f_schaeffler Follower: 20k Tweets: 1,092	@PetrBystronAFD Follower: 17k Tweets: 496	@SusanneHennig Follower: 29k Tweets: 4,463

Table 7: The 10 accounts with the most followers for each party (as of January 2022).

A.4 Word clouds for positive and negative tweets

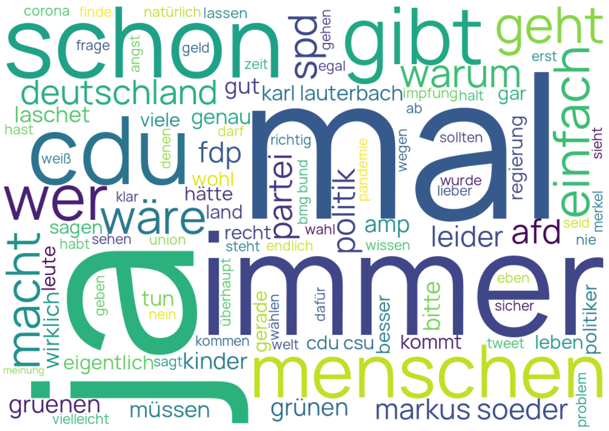


Figure 4: Word cloud created of negative tweets in the corpus for all parties.(These visualizations were generated by the Python package *wordcloud*.)

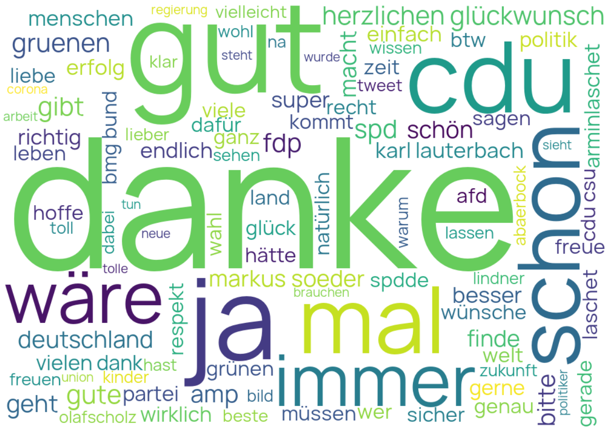


Figure 5: Word cloud created of positive tweets in the corpus for all parties.

“Japan’s Answer to Mozart”: Automatic Detection of Generalized Patterns of Vossian Antonomasia

Michel Schwab[1], **Robert Jäschke**[1,2] **and Frank Fischer**[3]
[1]Humboldt-Universität zu Berlin, Germany
[2]L3S Research Center, Hannover, Germany
[3]Freie Universität Berlin, Germany
{michel.schwab,robert.jaeschke}@hu-berlin.de
fr.fischer@fu-berlin.de

Abstract

Vossian Antonomasia (VA) is a rhetorical device used to describe an entity (the target) by transferring certain features and characteristics of another entity (the source) to it. The phenomenon is closely related to metaphor and metonymy. Similar to these more familiar devices, the detection of VA expressions is a challenging task. We propose novel VA detection models that center on the source to tackle this problem. The focus lies on the ability of the models to detect VA independent of the syntactic patterns they appear in. We model the problem in different scenarios and utilize a state-of-the-art metonymy resolution model that relies on word masking, and metaphor detection models, which are based on linguistic metaphor theories, and adjust them to our task. All models leverage pre-trained language models such as BERT and RoBERTa. As there is limited annotated data available, we use a data augmentation technique to create a new dataset consisting of VA with new syntactic patterns where the generalization ability of the models can be evaluated.

1 Introduction

Vossian Antonomasia (VA) is a stylistic device that refers to an entity by naming another famous named entity that shares certain characteristics or sets of attributes with the entity. In general, it consists of three chunks (Bergien, 2013): The *target* is the entity which is being described. The *source* is the famous entity that typically stands for a certain set of attributes. The *modifier* is the component that shifts the characteristics of the source to the target’s environment. When Angela Merkel is referred to as “the German Margaret Thatcher” (Trippe, 2005), “Angela Merkel” is the target entity that inherits one or more attributes from the source entity, in this case from the Iron Lady, “Margaret Thatcher”. The modifier (“German”) projects these attributes traditionally associated with Margaret Thatcher onto Angela Merkel. The combination of source and modifier is called a *VA phrase* in the following.

To understand VA, one requires a deep cultural and historical knowledge of the source entity, as the transferred characteristics are often not explicitly mentioned, but only indicated by the name of the source that stands for the attributes. Thus, the readers themselves must infer the author’s intention. This can be achieved by the context the expression appears in and knowledge about the source itself. The context is quite important because, in most cases, an entity does not only stand for one property. Arnold Schwarzenegger serves as a good example of a person who successfully moved between fields and changed the characteristics and attributes he stands for. First, he was known as a successful bodybuilder, but after turning to acting and politics, the focus of his persona shifted to his newly achieved accomplishments and his ability to successfully transition between fields.

The automatic detection of VA is challenging as their syntax is often ambiguous and hard to distinguish from literal expressions. See, for example, “the German Angela Merkel” vs. “the American Angela Merkel” (Pohl, 2016). The first phrase is literal stating that Angela Merkel is a person from Germany. In contrast, in the second phrase, Angela Merkel stands for a set of characteristics and is used as a source in a VA expression to describe Hillary Clinton.

Recent years have seen various approaches to the automatic detection and extraction of VA from larger text corpora. The first steps were pattern-based approaches (Jäschke et al., 2017; Fischer and Jäschke, 2019; Schwab et al., 2019), but recently language models like BERT (Devlin et al., 2019) were employed (Schwab et al., 2022). They achieved strong results and are also robust on unseen data.

In this paper, we tackle the problem of detecting VA expressions independent of their syntactic structure. The syntactic structure of VA phrases consisting of source and modifier can include a wide range of variations. So far, the only annotated VA dataset (Schwab et al., 2023b) consists solely of examples where the modifier follows the source, i.e., "the SOURCE of MODIFIER", which is a commonly used syntactic pattern for VA phrases. This is because of the variety of naming the modifier. In comparison to other syntactic patterns, the modifier in this pattern can have an arbitrary length and complex structure. In contrast, other patterns such as "the MODIFIER SOURCE" (see Table 1 for more syntactic patterns) often impose stricter limitations, typically requiring the modifier to be a single word, such as an adjective or noun. To our knowledge, there is no study describing the extraction of VA where the modifier precedes the source. We address the problem of a generalized VA detection approach by focusing solely on the source during training to remove the boundaries associated with the modifier and target. To achieve this, we develop five different methods. One is a sentence classification model that uses special tokens to indicate the candidates. The next is based on a sentence-pair classification model. Two rely on linguistic metaphor theories. We adapt the metaphor theories to the source of VA expressions, since source entities in text, like metaphorical words, are not meant literally. The last method is an adaptation of a metonymy resolution model, since VA is often categorized as a specific subtype of metonymy.

Next to getting a deeper understanding of the phenomenon itself, the detection of VA can support various NLP tasks. It can provide new and interesting question answering challenges, as Schwab et al. (2023a) have shown that one can easily transform the combination of source and modifier (VA phrase) into questions. Schwab et al. (2023a) also showed that VA phrases are hard to be captured correctly by coreference resolution models. The models must understand that the source is not independent, but a part of the target's reference chain. Often, this did not work and the sources were predicted in new standalone reference chains. Thus, VA detection could improve and support coreference resolution. By understanding figures of speech like VA, language models can better understand natural language in general and especially the nuances of human language. With that, more human-like text could also be generated, for instance, spiced-up headlines for newspaper articles.

This paper is structured as follows: In Section 2, we discuss related work, while in Section 3, we present the datasets and explain the dataset generation process in detail. In Section 4, we describe the developed models and methodology, followed by an empirical evaluation of the proposed models in Section 5. Finally, Section 6 closes the paper with a conclusion.

Our code and data are freely available.[1]

2 Related Work

The research on the automatic detection of VA is a relatively new topic in the NLP area. There exist multiple approaches on the (semi-)automatic detection and extraction of the phenomenon that have been developed recently. Jäschke et al. (2017), Fischer and Jäschke (2019) and Schwab et al. (2019) used semi-automatic approaches that were based on syntactic patterns around the source. In particular, they used regular expressions to extract candidate sentences from a newspaper corpus and matched those candidates against entity lists. Fischer and Jäschke (2019) and Schwab et al. (2019) removed common false positives in a second step using a manually curated blacklist. While Schwab et al. (2019) additionally presented a first fully automatic approach for VA detection employing a bidirectional long short-term memory (BLSTM) network, in Schwab et al. (2022) the approaches using neural networks were more advanced. They used concatenations of BLSTM and attention layers with ElMo embeddings (Peters et al., 2018) as well as a fine-tuned BERT model (Devlin et al., 2019) for binary sentence classification. Additionally, they presented a VA tagger that tags all parts of a VA expression in a sentence employing BLSTM and conditional random fields as well as a fine-tuned BERT model for sequence tagging. Schwab et al. (2023a) did not detect VA expressions, but tackled the task of detecting the target entity inside the newspaper article in which the VA expression appeared, which was neglected in the previous approaches. They showed that by transforming a VA phrase into a question, a hybrid model that sequentially uses a QA model and a coreference resolution model could yield high scores without fine-tuning the models further.

Similar tasks like metaphor detection have been studied deeply. Most of the research is based on

[1] https://vossanto.weltliteratur.net/icnlsp2023/

the Mozart of Japan
Japan's (the Japanese) Mozart
Japan's (the Japanese) answer to Mozart
Japan's (the Japanese) version of Mozart
Japan's (the Japanese) equivalent of Mozart

Table 1: An example of the data augmentation versions with nouns (Japan) and their adjective forms as modifiers in brackets (Japanese). In total, we get eight additional versions per sentence.

neural networks. While Gao et al. (2018), Dankers et al. (2019) and Torres Rivera et al. (2020) used sequence tagging models based on contextual word embeddings, other models are focusing on single word classification. In particular, they classify words according to whether they are meant literally or metaphorically. Choi et al. (2021) make use of two linguistic metaphor theories which they implement by employing a pre-trained language model, RoBERTa, and extract the embeddings in context and without context to train a multilayer perceptron (MLP). Most recently, Wang et al. (2023) follows the idea of Choi et al. (2021), but additionally focuses on selecting relevant context for the classification task employing a dependency parser for denoising the context around the candidate word which works especially well on long input sentences.

Metonymy resolution is another similar task that has been researched, especially recently with the use of pre-trained language models (Su et al., 2020; Li et al., 2020; Mathews and Strube, 2021). Often, the task is limited to location metonymy resolution (Li et al., 2020; Su et al., 2020). While Li et al. (2020) models the task as a token-level classification task, Mathews and Strube (2021) introduces a sequence tagging approach. Both models mask their candidates during training and evaluation.

3 Data

Candidate Generation We use the dataset from Schwab et al. (2023b) for training. There, we need to identify phrases that are candidates for VA sources. As, by definition, the source of any VA expression has to be a named entity, we utilize a state-of-the-art named entity tagger, FLAIR (Akbik et al., 2019), to obtain all candidate entities for each sentence in the dataset. We then collect tuples for each sentence consisting of an entity in the sentence and the sentence itself. The tuples containing a source entity are labeled positive, all others negative.

We remove candidates where the text sequence the NER tagger has identified as entity mention does not exactly match a source phrase, as those cases are difficult to handle correctly (and only a small part of the data is affected, cf. Sec. 5). Consider the sentence "He is the Michael Jordan of swimming, but he was never as good as Michel Jordan.". If the tagger would only tag "Michael" or "Michael Jordan of swimming" as an entity we remove those candidates.

The sentence highlights another issue: an entity that is mentioned more than once in the same sentence can not be distinguished within our set of tuples. When all mentions are no VA sources, we keep the tuples. When one of those entity mentions is indeed a source, we keep that tuple (i.e., the tuple with the positive label) and remove the others, since such cases are very rare. We removed 38 negative tuples in the training data and nine negative tuples in the test data.

Training Data Compared to other more popular rhetorical devices, such as metaphor and metonymy, one challenge of VA detection is the lack of annotated data. The only annotated English VA dataset is, to our knowledge, the one by Schwab et al. (2023b). It was first introduced by Schwab et al. (2019) and later annotated further (Schwab et al., 2022, 2023a). The dataset contains sentences from the New York Times Annotated Corpus (Sandhaus, 2008). The corpus contains articles from the New York Times from 1987 to 2007, comprising around 60,000,000 sentences. The dataset was created in a semi-automated way. First, frequently used syntactical patterns around the source were identified and candidate sentences extracted. The patterns consist of one of the words before (`the/a/an`) and after the source (`of/for/among`). Using all possible combinations, the authors of Schwab et al. (2019) obtained nine different patterns. Then the words between these combinations were matched against an entity list, and finally those candidates were checked against a manually curated black list to remove false positives and manually labeled. In total, the dataset contains 6,095 sentences of which 3,115 include VA expressions. On this dataset we generate candidates as explained before, which results in a training dataset of 16,877 sentence-entity tuples with 2,868 positive instances (17%) and 14,009 negative instances.

Test Data The lack of syntactic variations of VA expressions is one significant issue for testing VA detection models on generalization. For example, in the training data the modifier always appears directly after the source:

```
(a|an|the) SOURCE (of|for|among) MODIFIER.
```

This pattern, however, does not cover all variants of VA as the syntactic patterns in which source and modifier appear are more diverse. Annotating a text corpus to identify new syntactic variations is prohibitively expensive due to the rarity of the phenomenon on the sentence level (Schwab et al., 2019). Another approach is syntactic data augmentation which changes the syntax of a sentence without affecting its semantics. In our case, it is especially crucial to ensure that the VA expressions remain intact and that their meaning is not changed.

To augment data, we identified eight different VA patterns consisting of source and modifier that are different from the ones in the training data. In particular, their source follows *after* the modifier. Two of the patterns have no words between the modifier and source (named "—" in the sequel), the other six patterns have connecting words between modifier and source. We refer to these phrases ("answer to","version of", "equivalent of") as "connector phrase" (CP). Each of the phrases appears two times in the patterns.

The first four patterns are represented by the following regular expression and involve a modifier that is a noun:

```
MODIFIER's (CP)? SOURCE,
```

where CP is a connector phrase, MODIFIER is the modifier chunk and SOURCE is the source entity. The remaining four patterns include a modifier that is an adjective and are represented by the following regular expression:

```
(a|an|the) MODIFIER (CP)? SOURCE,
```

where the choice of the article at the beginning ("a", "an" or "the") depends on the article in the original VA phrase.

Six of the eight patterns include a CP between modifier and source consisting of two words, whereas the other two patterns do not have any words between them. This is another distinction from the pole word succeeding the source in the annotated data. Furthermore, the grammatical category of the modifier changes. While 92% of the modifiers in the annotated data are noun phrases,[2] this is not the case for the last four patterns, where the modifier is an adjective.

The modifiers in the annotated data can be complex (the longest modifier consists of 25 words) and not all VA phrases can be easily augmented as they cannot be changed semantically correct into a noun or adjective. Thus, we use a subset of the data for augmentation. Specifically, we extract all sentences that include a VA expression where the modifier is a geographical place that possesses an adjectival form. This ensures that the meaning is not changed when adapting the modifier and the syntax. We achieve this using the lists of adjectival and demonymic forms of place names from Wikipedia.[3] This has the advantage that those modifiers can always be transformed into any of the eight patterns since place names are always nouns and have an adjectival form which is suitable for both, noun and adjective modifiers.

Hence, we match all modifiers against the lists. In total, we could extract 244 VA expressions where the modifier matches an entry in one of the Wikipedia lists. Countries were mentioned most often (159), followed by cities (51), regions (23), and continents (11).

By augmenting each sentence, we obtain 1,952 augmented sentences (244 per pattern), which we call *augmented data*. Along with the 244 original instances (*original data*), this yields a total of 2,196 sentences. Again, we apply the candidate generation method and compute entity-sentence tuples for each sentence. In total, we produced 8,480 unique instances of which 2,196 are positive and 6,284 are negative. The positive label ratio increases compared to the positive label ratio of the training data to 26% as each sentence in the test data consists of a VA source which is not the case in the training data. Each sentence produced on average 34.5 instances with a standard deviation of 16.1 which shows that the number of generated instances per sentence is quite diverse depending on the number of named entities.

4 Methods

As explained before, the anchor of a Vossian Antonomasia expression is the source entity which is being invoked as a point of comparison. Thus, we

[2] Which we detected with the dependency parser from spaCy (Honnibal et al., 2020).

[3] https://en.wikipedia.org/wiki/List_of_adjectival_and_demonymic_forms_of_place_names

aim to find source entities of VA expressions using different approaches. In particular, our goal is to identify generalized VA expressions across diverse syntactic structures. As a baseline, we adapt a state-of-the-art VA extraction model. We then make use of models that were successfully applied in similar areas. In particular, we adjust two metaphor detection models which are based on linguistic metaphor theories. Similar to metaphorical words, the source entity of a VA expression is not meant literally, but stands for a set of characteristics. Additionally, we adapt a metonymy resolution model that uses candidate word masking. Finally, we present two fine-tuned RoBERTa models for sentence(-pair) classification which are designed to focus on the entity candidates inside a sentence. Contrary to the baseline, which is a sequence tagging model, all subsequent models are binary classification models. The goal is to determine whether pre-computed entities serve as a source of a VA expression or not. As explained in Section 3, we first identify all entities in a sentence and then classify each entity-sentence tuple.

BERT_SEQ This baseline is an adaptation of the sequence tagging model in Schwab et al. (2022), BERT_SEQ. Each word in a sentence is tagged to determine whether it is part of a chunk of a VA expression (i.e., target, source, or modifier) or not. It is a fine-tuned BERT (base-cased) model which outperformed a BLSTM-CRF model.

Here, we modify this model by focusing on tagging the source words only, employing the IOB tagging scheme. In particular, like Schwab et al. (2022), we add an additional linear layer to the BERT model to tag all words in a sentence. This enables a better comparability with our newly developed models. Additionally, the implicit focus on the order of chunks will vanish, which potentially leads to better generalization.

BERT_MASK This model is an adaptation of a state-of-the-art location metonymy resolution model (Li et al., 2020). The model is based on the idea that the context is more important to distinguish between metonymic and literal usage than the potential metonymic candidate itself. We use this model, since VA is often categorized as a subtype of metonymy, and apply it to our source candidates. In particular, we follow Li et al. (2020) in that we mask the source candidates with the single token *X* during training and evaluation. Further, we fine-tune a BERT model for word-level classification. Specifically, we extract the embeddings of the masked token (*X*) from the last hidden layer of BERT and feed them into a binary linear classifier to classify whether the candidate is a VA source. In cases where the masked token is omitted in the preprocessing due to truncation, we utilize the *[CLS]* token for classification instead.

RoBERTa_MIP This model is inspired by the Metaphor Identification Procedure (MIP) (Group, 2007). In short, the theory proposes that a word is used metaphorically when its literal meaning deviates from its contextual meaning. The key is that the literal meaning of the word would not apply directly in the used context. We transfer this concept to the source entity of a VA expression: The source entity is not meant literally, but is a placeholder for a set of characteristics the entity stands for. The entity would normally not be used in the context, as the target and especially the modifier are normally not directly related to the source. Thus, we state that a named entity used as a source in a VA expression has a different meaning in the context (a set of characteristics) from its more basic meaning (the entity itself).

We then roughly follow Choi et al. (2021). As base, we utilize RoBERTa (Liu et al., 2019), a pretrained language model. First, each word of a sentence is tokenized. The special character sequences `<s>` and `</s>` are then added at the beginning and end of the token sequence, respectively. Next, we compute the position embedding which represents the position of each token within the sentence and the segment embedding which indicates which tokens belong to the candidate. The input embeddings are finally obtained by the element-wise addition of token, position embedding, and segment embedding. In a separate step, we tokenize the isolated candidate words using the same tokenizer and also add special characters accordingly.

The embeddings of the candidate tokens in both steps are averaged independently. This results in two embeddings: the contextualized embedding and the isolated embedding for the candidate. These embeddings are then concatenated and passed through a linear layer that outputs a binary label indicating whether the entity is a VA source or not.

RoBERTa_SPV The model is based on Selectional Preference Violation (SPV) (Wilks, 1975, 1978), which is popular in metaphor detection methods, see Mao et al. (2019) and Choi et al. (2021). The idea of SPV in the context of metaphors is that a word is metaphorical when it appears unusual in its surrounding context, that is, it typically does not co-occur with the surrounding words. We adapt this theory to VA detection: An entity serving as a source for VA expressions appears unusual within its surrounding words, especially with the modifier which normally represents an environment unrelated to the source. Instead, the modifier is connected to the target entity.

As in the MIP model, we compute tokens, position embedding and segment embedding of the sentence. Subsequently, we compute the contextualized embedding accordingly and also calculate the embedding of the special <s> token, which represents the aggregated representation of the sentence. Both embeddings are concatenated and passed through a linear layer which returns a binary label.

RoBERTa_CLF We adapt the binary sentence classification model from Schwab et al. (2022). Specifically, we introduce two special tokens, `[START_SRC]` and `[END_SRC]`, to denote the start and end, respectively, of the candidate by encasing the source entity inside the sentence with both tokens. These tokens are added to the tokenizer. The adapted sentence is then used as input RoBERTa which we fine-tune for binary sentence classification by adding and training a linear layer.

RoBERTa_PAIR For this model, we reformulate the task as a sentence-pair classification problem. This task is typically used to assess the relationship between two sentences, such as next sentence prediction, contradiction of sentence-pairs or semantic relations. In our case, the first sentence consists of the candidate entity only, while the second sentence provides context in form of the corresponding sentence the candidate entity appears in. We want the model to learn to classify whether the candidate entity is a source in the corresponding sentence or not. As in RoBERTa_CLF, we adapt RoBERTa by appending a linear layer on top of the RoBERTa model. Subsequently, we fine-tune the model for binary classification.

5 Evaluation

In this section, we describe the experimental settings before presenting and analyzing the empirical results of our models. All models rely on the output of an NER tagger whose output is used to form the set of source candidates. This is different from the baseline model, which does not need candidates but classifies each word individually.

The tagger we used in our study has an F_1 of 0.94 on the CoNLL-03 dataset. In our specific case, it missed identifying 110 (3.8%) out of 2,868 source entities in the training dataset and 40 (1.8%) out of 2,196 source entities in the test dataset. For the sake of comparability, we exclude these instances from the evaluation process. The idea behind the exclusion is that we aim to evaluate the individual performance of our models rather than the whole performance of the NER tagger combined with our models. We use precision, recall and F_1 score to assess the performance of the models.

5.1 Experimental Settings

We conduct hyperparameter optimization on dropout rate, epochs, learning rate, and batch size based on F_1 score.[4] For this, we use 25% of the test data as a validation set for all models including the baseline. We use a part of the test data for hyperparameter optimization on purpose as we want to determine the best model for the generalized test data. We assume that if the model works well on the test data, it should still be able to achieve good performance on the data we trained it with. We will evaluate this in the subsequent section.

We use the pre-trained BERT base-cased model[5] as in Schwab et al. (2022) for BERT_SEQ as well as for BERT_MASK, and the pre-trained RoBERTa base model[6] for all other models as basis. Both models share the same architectural parameters. Specifically, each model has 12 transformer blocks, 12 attention heads, and the dimensionality of the hidden states is set to 768. For all models, we use AdamW optimizer (Loshchilov and Hutter, 2017). We implemented our models using the Hugging Face framework and PyTorch. The code is free available on our website.[7]

[4] See Appendix A for details and final choices for each model.

[5] https://huggingface.co/bert-base-cased

[6] https://huggingface.co/roberta-base

[7] https://vossanto.weltliteratur.net/icnlsp2023/

model	precision	recall	F_1
BERT_SEQ	.88 ±.02	**.97** ±.03	**.92** ±.02
BERT_MASK	.83 ±.03	.88 ±.02	.85 ±.02
RoBERTa_MIP	.88 ±.02	.89 ±.04	.88 ±.01
RoBERTa_SPV	**.93** ±.03	.85 ±.07	.89 ±.03
RoBERTa_CLF	.87 ±.03	.87 ±.02	.87 ±.02
RoBERTa_PAIR	.76 ±.04	.94 ±.02	.84 ±.01

Table 2: Performance of the models using 5-fold cross validation on the training dataset.

5.2 Results on Training Data

Table 2 presents the results on the training data using stratified 5-fold cross validation. All approaches, even if hyperparameters were not optimized for this data, achieve strong results. Surprisingly, the baseline, BERT_SEQ, has the best results, having an F_1 score of 0.92, although the gap to the other models is not large. RoBERTa_CLF and both adapted metaphor detection model, RoBERTa_MIP and RoBERTa_SPV, have similar scores of 0.87, 0.88, and 0.89, respectively. Only BERT_MASK and RoBERTa_PAIR achieve a little lower score of 0.85 and 0.84, respectively. The general high scores are expected as the models were trained on similar data regarding the syntax of the VA expressions. Also, the label ratio is the same as we conducted stratified sampling for the cross validation.

5.3 Zero-shot Results on Test Data

We conduct a zero-shot transfer with our models on the test data consisting of the original and augmented data as explained in Section 3. This evaluation is conducted to analyze how the models generalize to new syntactic VA variations which is the main goal of our work. In this evaluation, we obtain surprising results. While the performance of all models except RoBERTa_PAIR decreases drastically in all metrics, RoBERTa_PAIR increases its performance to an F_1 of .86 (cf. Table 3). While the precision of RoBERTa_PAIR increases substantially, the recall decreases. The other models are not able to compete against this model. Even the results of the second best model, RoBERTa_MIP, decreases to an F_1 score of 0.74 which is a gap of 0.12 points. Still, this is the smallest performance gap and shows that the adaptation of the MIP theory works better for generalized VA detection than the rest of the models. BERT_MASK attains the lowest F_1 score of 0.25. The baseline, which achieved the best results on the training data, is also not able to solve this task with the second lowest F_1 of 0.27 as well as RoBERTa_SPV and RoBERTa_CLF whose scores also dropped to 0.58 and 0.41, respectively.

The performance on the original data in the test dataset (713 instances, 183 positive) even increases for all models compared to the results on the training data. This is not that surprising, as the syntax is the same as in the training data. Also, the training data consists of sentences without VA expressions that are syntactically very similar to those with VA expressions. In the test data, however, there exist no such negative examples. Thus, this might be a reason why the scores are rising. The performance on the augmented data drops dramatically in almost all models compared to the performance on the training data (cf. Section 5.2).

This shows that only RoBERTa_PAIR is able to handle new syntactic variations in contrast to all other models. As the F_1 almost did not change between the evaluation on both datasets, it shows a robustness to new data. The metaphor detection models had similar scores on the training data and could obtain high scores on the original data, but they diverge on the augmented data. While RoBERTa_SPV drops to an F_1 of 0.53, which is 0.36 points less than on the training data, the performance gap of RoBERTa_MIP is smaller.

In general, it seems that in all models except RoBERTa_PAIR, the syntax of the VA expression still has a major influence on the correct classification.

Performance vs. Syntax (RoBERTa_PAIR) An interesting point to investigate further is the influence of the syntactic variations we used for data augmentation. In total, we have four syntactic patterns, three that consist of the connector phrases between modifier and source, “answer to”, “version of”, “equivalent of”, and the “—” version without any connector phrase between both chunks. Table 4 shows the results in the ‘total’ block. We can see that all three metrics are best for the pattern “—”, with an F_1 of 0.92. For the “equivalent of” and “version of” patterns, the model still achieves high scores, whereas for the “answer to” patterns, it performs worse with an F_1 of 0.72 which is 0.2 lower than the best score. It is interesting that one pattern is much harder to detect and shows that even if patterns seem quite similar for humans, it is much

	total			original data			augmented data		
model	precision	recall	F_1	precision	recall	F_1	precision	recall	F_1
BERT_SEQ	.73	.17	.27	**.99**	**1.00**	**.99**	.42	.07	.12
BERT_MASK	.69	.15	.25	.92	.81	.86	.52	.07	.13
RoBERTa_MIP	.92	.62	.74	.97	.92	.95	.91	.58	.71
RoBERTa_SPV	**.97**	.42	.58	**.99**	.87	.93	**.97**	.36	.53
RoBERTa_CLF	.73	.29	.41	.95	.82	.88	.66	.22	.33
RoBERTa_PAIR	.89	**.83**	**.86**	.91	.97	.94	.89	**.81**	**.85**

Table 3: Performance of the models on the test data which include the original and augmented data.

	total			modifier is a noun			modifier is an adjective		
syntax	precision	recall	F_1	precision	recall	F_1	precision	recall	F_1
—	**.91**	**.93**	**.92**	**.90**	**.93**	**.92**	**.91**	**.93**	**.92**
answer to	.85	.62	.72	.88	.72	.79	.82	.51	.63
version of	.89	.83	.86	**.90**	.90	.90	.87	.77	.81
equivalent of	.89	.87	.88	**.90**	.91	.91	.87	.82	.85
total	.89	.81	.85	.90	.87	.88	.87	.76	.81

Table 4: Performance of RoBERTa_PAIR on the augmented data, split up by pattern and POS type.

harder for the models to detect them correctly.

Performance vs. POS (RoBERTa_PAIR) We now analyze whether the part of speech (POS) tag of the modifier influences the model using the best performing model on the test dataset, RoBERTa_PAIR. One half of the augmented data has modifiers that are adjectives whereas the other half has modifiers that are nouns. In Table 4, we can clearly see that the model performs better when the modifier is a noun with an F_1 of 0.88 compared to 0.81 for the adjective examples. One reason is the high performance gap of 0.11 in recall. A plausible reason is the fact that in the training data, the modifiers of the VA expressions are almost always noun phrases (and thus include at least a noun), which possibly is captured in the fine-tuning process, even if the modifier is not marked explicitly.

5.4 Error Analysis (RoBERTa_PAIR)

In total, we got 851 false positive and 159 false negative errors in the 5-fold cross validation. In 239 cases, an entity candidate was falsely predicted as source entity in a sentence that included a VA expression. Still, in the majority (612) of the false positive errors, the entities appeared in a sentence without any VA occurrence.

In the test dataset, more false negative errors (281) than false positives (172) occurred. Grouping the false positives by entity and original sentence, we got 25 groups where in 14 of them all augmentations with the same candidate were predicted falsely. The false negatives, on the other hand, grouped into 80 groups, which makes sense as the syntax around the source entities changed, whereas the syntax around the entity candidates in the false positive instances did not and thus, the model's prediction should be more similar.

Table 5 shows a sample of false positive and false negative errors, the RoBERTa_PAIR model did in the test dataset.

The false positives included candidate entities that belonged to the VA expression but as a target chunk ("Manno Charlemagne") or as a modifier chunk ("European"). That was expected as they are somehow connected semantically to the source and thus, it is harder for the model to differentiate between them. It also appeared that an entity that was used as source ("Berlusconi") was also mentioned in another typing elsewhere in the sentence with a literal meaning ("Silvio Berlusconi"). Those examples are rare as the source is normally not mentioned in the context. Still, these are decisions that are especially hard to predict correctly for the sentence pair model as the model has no explicit focus on the position of the entity in the sentence as the other models had.

Ex. 1: He doesn't want to be Syria's version of **Gorbachev**.

Ex. 2: "He's the Japanese answer to **Cal Ripken**, but with more punch," said Marty Kuehnert, a sports broadcaster and longtime resident in Japan.

Ex. 3: Buena Vista Home Entertainment, the distribution arm of Disney, recently acquired a library of Japanimation created by a man often hailed as "the **Walt Disney** of Japan," Hiyao Miyazaki.

Ex. 4: One of the anthology's strongest cuts, "Ayiti Pa Fore" ("Haiti Is Not a Forest') was recorded in 1988 and features **Manno Charlemagne**, a singer and songwriter who is regarded as Haiti's answer to Bob Marley.

Ex. 5: In the capital, intellectuals refer to Mr. Thaksin as Asia's Berlusconi, a reference to Prime Minister **Silvio Berlusconi** of Italy, a business tycoon who has faced continuing accusations of conflict of interest.

Ex. 6: Its chairman, Jan Carlzon, is credited with turning the airline around in the early 1980s, earning a reputation as "the **European** answer to Lee Iacocca," one analyst said.

Table 5: Incorrectly classified instances of RoBERTa_PAIR on the test dataset. False negatives (Ex. 1-3) are marked green, false positives (Ex. 4-6) red.

6 Conclusion

We proposed four novel VA detection models and analyzed their ability to detect generalized VA expressions across a range of syntactic patterns. To achieve this, we use data augmentation techniques to create a VA dataset including numerous new syntactic patterns. We develop VA detection models based on adjusted linguistic metaphor theories and a metonymy resolution model that are applied to the source. While most models struggle to generalize well to these new patterns, our best model, RoBERTA_PAIR, achieves good results on both, the training and test dataset.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Angelika Bergien. 2013. Names as frames in current-day media discourse. In *Name and Naming. Proceedings of the second international conference on onomastics*, pages 19–27, Cluj-Napoca. Editura Mega.

Minjin Choi, Sunkyung Lee, Eunseong Choi, Heesoo Park, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee. 2021. MelBERT: Metaphor detection via contextualized late interaction using metaphorical identification theories. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1763–1773, Online. Association for Computational Linguistics.

Verna Dankers, Marek Rei, Martha Lewis, and Ekaterina Shutova. 2019. Modelling the interplay of metaphor and emotion through multitask learning. 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 2218–2229, Hong Kong, China. 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.

Frank Fischer and Robert Jäschke. 2019. 'The Michael Jordan of greatness'—Extracting Vossian antonomasia from two decades of The New York Times, 1987–2007. *Digital Scholarship in the Humanities*, 35(1):34–42.

Ge Gao, Eunsol Choi, Yejin Choi, and Luke Zettlemoyer. 2018. Neural metaphor detection in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 607–613, Brussels, Belgium. Association for Computational Linguistics.

Pragglejaz Group. 2007. Mip: A method for identifying metaphorically used words in discourse. *Metaphor and symbol*, 22(1):1–39.

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

Robert Jäschke, Jannik Strötgen, Elena Krotova, and Frank Fischer. 2017. “Der Helmut Kohl unter den Brotaufstrichen”. Zur Extraktion Vossianischer Antonomasien aus großen Zeitungskorpora. In *Proceedings of the DHd 2017*, DHd ’17, pages 120–124. Digital Humanities im deutschsprachigen Raum.

Haonan Li, Maria Vasardani, Martin Tomko, and Timothy Baldwin. 2020. Target word masking for location metonymy resolution. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3696–3707, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Rui Mao, Chenghua Lin, and Frank Guerin. 2019. End-to-end sequential metaphor identification inspired by linguistic theories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3888–3898, Florence, Italy. Association for Computational Linguistics.

Kevin Alex Mathews and Michael Strube. 2021. Impact of target word and context on end-to-end metonymy detection.

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.

Ines Pohl. 2016. "America’s fear has a new name: merkel". Accessed on 06 23, 2023.

Evan Sandhaus. 2008. The New York Times Annotated Corpus LDC2008T19. DVD, Linguistic Data Consortium, Philadelphia.

Michel Schwab, Robert Jäschke, and Frank Fischer. 2022. "The Rodney Dangerfield of Stylistic Devices": End-to-end detection and extraction of vossian antonomasia using neural networks. *Frontiers in artificial intelligence*, 5.

Michel Schwab, Robert Jäschke, and Frank Fischer. 2023a. “who is the madonna of Italian-American literature?”: Target entity extraction and analysis of vossian antonomasia. In *Proceedings of the 7th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 110–115, Dubrovnik, Croatia. Association for Computational Linguistics.

Michel Schwab, Robert Jäschke, Frank Fischer, and Jannik Strötgen. 2019. “a buster keaton of linguistics”: First automated approaches for the extraction of vossian antonomasia. 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 6238–6243, Hong Kong, China. Association for Computational Linguistics.

Michel Schwab, Robert Jäschke, and Frank Fischer. 2023b. Annotated vossian antonomasia dataset.

Chuandong Su, Xiaoxi Huang, Fumiyo Fukumoto, Jiyi Li, Rongbo Wang, and Zhiqun Chen. 2020. English and chinese neural metonymy recognition based on semantic priority interruption theory. *IEEE Access*, 8:30060–30068.

Andrés Torres Rivera, Antoni Oliver, Salvador Climent, and Marta Coll-Florit. 2020. Neural metaphor detection with a residual biLSTM-CRF model. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 197–203, Online. Association for Computational Linguistics.

Christian Trippe. 2005. "change is needed to get germany moving". Accessed on 06 23, 2023.

Shun Wang, Yucheng Li, Chenghua Lin, Loic Barrault, and Frank Guerin. 2023. Metaphor detection with effective context denoising. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1404–1409, Dubrovnik, Croatia. Association for Computational Linguistics.

Yorick Wilks. 1975. A preferential, pattern-seeking, semantics for natural language inference. *Artificial intelligence*, 6(1):53–74.

Yorick Wilks. 1978. Making preferences more active. *Artificial intelligence*, 11(3):197–223.

A Appendix

Hyperparameter optimization We conducted hyperparameter optimization using grid search on all models using F_1 score and a validation dataset that is 1/4 of the proposed test data, as explained in Section 3. The hyperparameters were tuned over the values given in Table 6. The values that we finally used for our models are given in Table 7.

hyperparameter	tested values
number of epochs	2, 3, 4, 5
batch size	8, 16, 32
maximal length	32, 64, 128
learning rate	$10^{-5}, 3 \cdot 10^{-5}, 5 \cdot 10^{-5}$
dropout rate	0.1, 0.2

Table 6: Values used for hyperparameter optimization.

model	epochs	batch size	max length	learning rate	dropout
BERT_SEQ	2	16	64	10^{-5}	0.2
BERT_MASK	4	16	32	$3 \cdot 10^{-5}$	0.2
RoBERTa_MIP	4	32	32	10^{-5}	0.2
RoBERTa_SPV	4	32	64	10^{-5}	0.2
RoBERTa_CLF	5	32	32	$3 \cdot 10^{-5}$	0.2
RoBERTA_PAIR	4	32	32	10^{-5}	0.2

Table 7: Final choice of model parameters.

GAVI: A Category-Aware Generative Approach for Brand Value Identification

Kassem Sabeh
Free University of Bozen-Bolzano
ksabeh@unibz.it

Mouna Kacimi
Wonder Technology Srl
mouna@wonderflow.ai

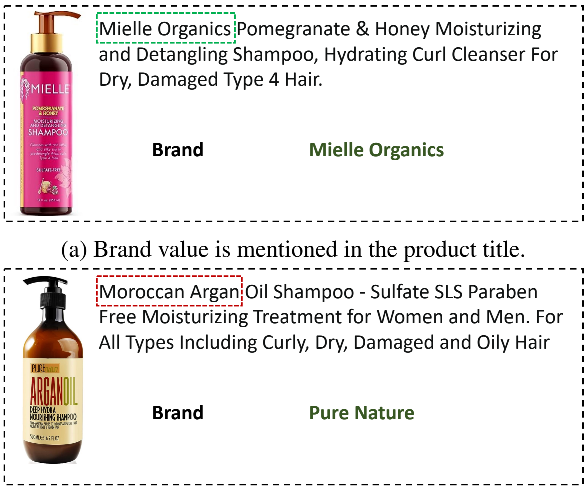
Johann Gamper
Free University of Bozen-Bolzano
jgamper@unibz.it

Abstract

Extracting product attribute value information is vital for many e-commerce applications. One of the most crucial product attributes is the brand, as it significantly impacts customers' purchasing decisions and behaviour. Consequently, it is critical for e-commerce platforms to automatically and accurately identify brand values from product descriptions. Most existing methods focus on brand value extraction from text descriptions using sequence tagging and question answering techniques. However, brand values are often not mentioned explicitly in the product descriptions. Also, these approaches are designed without paying attention to product categories, which are important for brand value identification. In this work, we propose a novel category-aware generative approach for brand value identification (GAVI). In particular, we formulate the brand value identification problem as a sequence-to-sequence generation task. We use the T5 language model as the backbone of our approach. This allows us to identify brand values that are not explicitly mentioned in the title in a generative manner. We then propose to highlight the product categories inside our model input, making the approach category-aware. We conduct extensive experiments on a public dataset for brand value identification. The experimental results demonstrate that our generation-based approach outperforms existing extraction-based methods. Our code is released along with the fine-tuned models presented in the paper[1], which are also available as a demo[2].

1 Introduction

Product attributes are a crucial component of e-commerce platforms as they provide valuable information for customers to browse and compare products. One of the most important product attributes is the brand, as it plays a pivotal role in influencing customers' behaviour and purchasing decisions (Chovanová et al., 2015; Shahzad et al., 2014). Brand names also increase the recognisability of products and services amongst consumers, and permit them to deduce knowledge about important features of the product (Zhang et al., 2015). For instance, Figure 1a shows an example of a profile of a shampoo product taken from an e-commerce website. The brand of this product is "Mielle Organics". Knowledge about the brand can help the customers build a set of associations, like this shampoo is of "*high quality natural and organic ingredients*", is tailored for "*frizzy or curly hair*", has a "*Moisturizing and Detangling effect*" and is designed "*for women*". Consequently, when customers shop for a shampoo, they often select a particular brand based on their prioritized attributes and features. These inherent correlations between brands and product attributes underscore the critical need for e-commerce applications to automatically and accurately identify brand names from product descriptions.


Mielle Organics Pomegranate & Honey Moisturizing and Detangling Shampoo, Hydrating Curl Cleanser For Dry, Damaged Type 4 Hair.
Brand
Mielle Organics
Moroccan Argan Oil Shampoo - Sulfate SLS Paraben Free Moisturizing Treatment for Women and Men. For All Types Including Curly, Dry, Damaged and Oily Hair
Brand
Pure Nature

(a) Brand value is mentioned in the product title.

(b) Brand value can not be extracted from the product title.

Figure 1: Examples of brand values in two product profiles.

[1] https://github.com/kassemsabeh/gavi
[2] https://bit.ly/3FHZGjU

Existing work for brand value identification falls under the general problem of attribute value extraction from product titles, with a plethora of research being developed to tackle this problem (Putthividhya and Hu, 2011; Kozareva et al., 2016; Zheng et al., 2018; Xu et al., 2019; Wang et al., 2020). Early approaches for attribute value extraction rely on rule based techniques and domain-specific dictionaries (Ghani et al., 2006; Vandic et al., 2012; Kozareva et al., 2016). These methods carry a close-world assumption and do not work well with new values, since they need to develop rules for every possible value. Consequently, they are not suitable for brand value identification where new brands are constantly emerging. With the advent of natural language understanding, sequence labeling methods have been developed (Huang et al., 2015; Zheng et al., 2018; Sabeh et al., 2022b). These methods utilize a BiLSTM-CRF architecture similar to NER tasks. However, their performance on attribute value extraction is limited by the abundance of negative token labels (e.g., the 'O' in BIO schema), which leads to many false negative results. Recently, question answering (Xu et al., 2019; Wang et al., 2020; Yang et al., 2022) based approaches were proposed. These methods scale existing sequence based methods to deal with multiple attribute inputs. All of the above approaches achieve promising results, however, they suffer from two major limitations:

- Most of the existing approaches are extractive-based methods; i.e., they extract the brand values from the text descriptions in the product profile. However, target brand values are sometimes absent from the textual descriptions of the product. For example, in Figure 1b, the brand of the product is "Pure Nature", which is not explicitly mentioned in the product title. The existing models extract instead the value "Moroccan Argan" as the brand, which is the wrong value.

- Existing methods for brand value extraction are designed without considering the product category. This is crucial for determining the set of applicable values, because categories can be substantially different in terms of brand values. For example, the value "Sunflower" can be a brand in the *Clothing* category. However, it used to indicate the scent in the *Food* category.

In this paper, we formulate the brand value identification problem as a sequence-to-sequence generation task. Inspired by the recent advances on text generation (Ushio et al., 2022; Bao et al., 2020; Xiao et al., 2021), we propose a category-aware **G**enerative **A**pproach for brand **V**alue **I**dentification, namely GAVI. In contrast to previous extractive approaches, we employ T5 (Raffel et al., 2020) language model as the backbone of our sequence-to-sequence approach. This generative approach allows the extraction output to expand beyond strings and sub-strings mentioned in the product textual description, which addresses the first limitation. To make the model category-aware, we propose to highlight the product categories inside the model input. This setup fits naturally with the sequence-to-sequence model architecture, and allows us to learn category-specific token embeddings that are effective for our task. We summarize the main contributions of our work as follows:

- We propose GAVI, a generative sequence-to-sequence model to identify brand values from product descriptions. To the best of our knowledge, this is the first work for generative brand value identification.

- We extend the basic generative solution to a category-aware sequence-to-sequence model by highlighting the product categories inside the input.

- We conduct extensive experiments on a public dataset, demonstrating the effectiveness of the proposed approach over several state-of-the-art baselines.

2 Related Work

Early work on attribute value extraction relied on rule-based techniques (Nadeau and Sekine, 2007; Vandic et al., 2012; Gopalakrishnan et al., 2012), which utilize domain-specific seed dictionaries to perform the extraction. After that, a myriad of studies formulated the extraction task as named entity recognition (NER) (Putthividhya and Hu, 2011; Bing et al., 2012; Ling and Weld, 2021; More, 2016). However, these approaches carry a closed world assumption and therefore can not discover new values of attributes.

With the advent of deep learning, a number of sequence tagging methods were proposed (Kozareva et al., 2016; Huang et al., 2015; Zheng et al., 2018).

These approaches make instead an open world assumption to discover new attribute values. (Huang et al., 2015) applied a BiLSTM-CRF model in a sequence tagging setting. (Zheng et al., 2018) developed an end-to-end tagging model (OpenTag) that benefits from an attention layer (Vaswani et al., 2017) to generate interpretable results. Moreover, (Xu et al., 2019) proposed to encode both attributes and values by using one set of BIO tags to scale up the tagging methods. (Karamanolakis et al., 2020) proposed a taxonomy aware multi-task framework that utilizes the taxonomy of the products to further improve the extraction. (Yan et al., 2021) utilize a hypernetwork (Ha et al., 2017) and Mixture-of-Experts module to parameterize their model with pre-trained attribute embeddings. (Sabeh et al., 2022b) proposed to utilize character level representations to improve the generalization performance of sequence tagging models for extracting brand values from product descriptions. The latest approaches (Wang et al., 2020; Yang et al., 2022; Sabeh et al., 2022a) reformulate the problem as a question answering (QA) task by utilizing BERT (Devlin et al., 2019), which allows them to scale to a large number of attributes. Sequence tagging approaches (Huang et al., 2015; Zheng et al., 2018; Sabeh et al., 2022b) are most relevant to our work because identifying brand names does not require scalability. However, these models are extractive and therefore can not infer brand names which are not directly mentioned in the title or description. They also fail to take product categories into account, which is crucial for brand value identification.

In this work, we adopt a generative approach to identify the brand values from the product descriptions. Our approach allows us to decode brand values that are not directly stated in the text descriptions. Our model is also category-aware, which allows us to effectively take the product categories into account.

3 Proposed Method

As mentioned above, previous methods formalize the brand value identification as a sequence tagging task. These approaches fail to identify brand values that are not explicitly mentioned in the product description. In this work, we tackle the task of brand value identification in a generative manner. More specifically, we propose to fine-tune a generative language model by formulating the brand value identification problem as a sequence-to-sequence generation task.

3.1 Problem Definition

In this section, we formally define the problem of brand value identification from the product description. Given an input product title $t = \{t_1, t_2, \ldots, t_n\}$ where n is the number of tokens in t. We refer to the product category as $c = \{c_1, c_2, \ldots, c_m\} \in C$, where m is the number of tokens in c, and C is a predefined set of categories. The goal of brand value identification is to generate a target sequence $\hat{v}$, which represents the target brand value. For the example in Figure 1a we have:

- t = "Mielle Organics Pomegranate & Honey Moisturizing and Detangling Shampoo, Hydrating Curl Cleanser For Dry, Damaged Type 4 Hair."
- c = "Shampoos"

To generate the value $\hat{v}$ given the product title t and the category c, we formulate the problem as a conditional sequence generation task. Formally, we optimize the model to maximize the conditional log-likelihood $P(v \mid t, c)$ as follows:

$$\hat{v} = \arg\max_{v} P(v \mid t, c)$$

In our implementation, similar to other sequence-to-sequence learning settings (Sutskever et al., 2014), we factorize the log-likelihood into word and sub-word level predictions.

3.2 Language Model Fine-tuning

We employ T5 (Raffel et al., 2020) sequence-to-sequence language model as the backbone of our approach. T5 is a tranformer-based (Vaswani et al., 2017) pre-trained generative language model that maps a given input sequence into an output sequence. The pre-trained T5 model achieves superior performance in many sequence-to-sequence tasks (Qi et al., 2020; Iqbal and Qureshi, 2022). Fine-tuning T5 language model for brand value identification can be done in a similar fashion as for sequence-to-sequence generation tasks, such as machine translation or text summarization, where the model generates a sequence of tokens given the input tokens (Dong et al., 2019; Bao et al., 2020; Xiao et al., 2021).

To make the model aware of the product category c, we propose to concatenate the input title

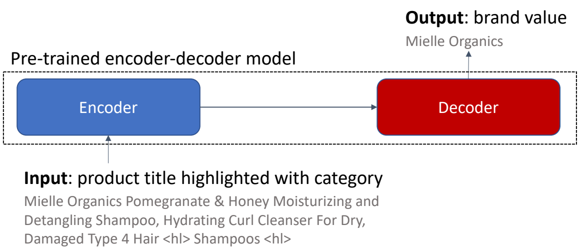

Output: brand value
Mielle Organics
Pre-trained encoder-decoder model
Encoder
Decoder
Input: product title highlighted with category
Mielle Organics Pomegranate & Honey Moisturizing and Detangling Shampoo, Hydrating Curl Cleanser For Dry, Damaged Type 4 Hair <hl> Shampoos <hl>

Figure 2: Overview of our generative approach GAVI; it takes category highlighted product title as input and returns the brand value. In this example, the model generates *Mielle Organics* as output.

t and category c into a single input x. After that, we highlight the category in the input. Specifically, following (Chan and Fan, 2019), we introduce a highlight token <hl> to take into account the category c inside the model input x as below:

$$x = \{t_1, t_2, \ldots, t_n, \texttt{<hl>}, c_1, c_2, \ldots, c_m\texttt{<hl>}\}$$

We could also choose not to include and highlight the category in our input. This means that we can also train a generative model that is not category aware by using only the title in our input x:

$$x = \{t_1, t_2, \ldots, t_n\}$$

In our experiments, we investigate and analyse these model variations, but assume the category highlighted title as the default input. We refer to the proposed category aware implementation of the T5 generative model as GAVI in our experiments. Figure 2 shows the overall architecture of our sequence-to-sequence generative approach.

4 Experimental Setup

In this section, we represent the experimental settings of our empirical approach for comparing our generative proposed models with state-of-the-art baselines on the task of brand value identification.

4.1 Datasets

We evaluate our model on a public product dataset[3] for brand value identification (Sabeh et al., 2022b). This dataset comprises over 250k product titles containing more than 50k unique brand values, derived from the Amazon Review Dataset (Ni et al., 2019). Each example consists of product title, product category, and the target brand value for identification.

Category	Number of Samples	Average Tokens
Grocery & Gourmet Food	22397	23.22
Toys & Games	63304	21.95
Sports & Outdoors	54214	21.57
Electronics	47870	32.17
Automotive	66837	23.75
Clothing, Shoes & Jewelry	85068	20.75
Pet Supplies	10868	23.72
Cell Phones & Accessories	78564	34.62

Table 1: Detailed statistics of the dataset. We use T5 tokenizer to tokenize the examples.

Category	Train	Val	Test
Grocery & Gourmet Food	15679	2239	4479
Toys & Games	44314	6330	12660
Sports & Outdoors	37951	5421	10842
Electronics	33512	4787	9574
Automotive	45132	6447	12894
Total	176588	25224	50449

Table 2: Statistics of AZ-base dataset with five selected categories.

Table 1 shows the statistical details of the dataset. The dataset contains information about products in eight main categories. The average number of tokens per sample in each category is also shown in Table 1. Following previous work (Sabeh et al., 2022b), we arrange the following setups for benchmark:

- **AZ-base** This split of the dataset contains information about products in five main categories: *Grocery & Gourmet Food, Toys & Games, Sports & Outdoors, Electronics* and *Automotive*. In this dataset, we randomly select 70% of the data for training, 20% for validation, and 10% for testing. The main purpose of this dataset is to evaluate the baseline model performance on the task of brand value identification. The statistics of the AZ-base dataset are provided in Table 2.

- **AZ-zero-shot** In order to evaluate the generalization ability of the models, we divide the AZ-base dataset into another disjoint training and test split with no overlapping brand

Category	Train	Val	Test
Clothing, Shoes & Jewelry	0	0	85068
Pet Supplies	0	0	10868
Cell Phones & Accessories	0	0	78564
Total	0	0	174500

Table 3: Number of samples in AZ-new-cat dataset.

[3] https://github.com/kassemsabeh/open-brand.

values. The test set of this split contains 8k unique values. None of these values are seen during training. This allows us to evaluate the zero-shot performance of the models.

- **AZ-new-cat** In this benchmark, we test the models ability in identifying brand values from different product categories. In specific, we use the same training set from AZ-base, but we test the model on three new categories of products. None of these categories are present in the training set, as shown in Table 3.

4.2 Implementation Details

All models are implemented using PyTorch[4], and are trained on NVIDIA Tesla V100 GPUs. During training, Adam (Kingma and Ba, 2015) optimizer is applied with initial learning rate $4e^{-5}$. The backbone uses the pre-trained T5-base encoder with 12 layers and 12 heads, which has 220M parameters. The embedding dimension is 768, while the maximal input length is set to 512. The batch size is set to 32. All hyper-parameters are chosen optimally based on the performance on the validation set of our dataset. We fine-tune the model on the training set for 10 epochs, and perform early stopping if there is no improvement in the loss on the validation set for 3 epochs. We report our final results on the test where we perform beam search of size four.

4.3 Evaluation Metrics

Following the literature (Xu et al., 2019; Yan et al., 2021; Wang et al., 2020), we use Precision (P), Recall (R), and F_1 as evaluation metrics. We compute these metrics based on the number of true positives (TP), false positives (FP), and false negatives (FN) of our predictions. We use Exact Match (Rajpurkar et al., 2016) criteria in our evaluations, where the full predicted sequence should match the ground truth.

$$P = \frac{TP}{TP + FP} \quad R = \frac{TP}{TP + FN} \quad F_1 = 2 \times \frac{P \times R}{P + R}$$

4.4 Compared Models

We compare the following models on the task of brand value identification:

BiLSTM-CRF (Huang et al., 2015) applies a BiLSTM followed by a CRF layer to model the dependency of the predicted tags.

OpenTag (Zheng et al., 2018) introduces a self-attention layer between the BiLSTM and CRF to highlight important features in the input. OpenTag is considered as the pioneer sequence tagging model for attribute value extraction.

OpenBrand[5] (Sabeh et al., 2022b) leverages a CNN encoder to generate character level representations and improve the generalization performance. OpenBrand achieves state-of-the-art results on the brand value extraction task.

T5 The base generative model of ours that is fine-tuned on the training dataset. T5 is not category-aware as it only uses the title in the input.

GAVI Our proposed category-aware generative model. Our model uses category-highlighted inputs to identify the brand values, as described in Section 3.2.

5 Experimental Results

In this section, we conduct a series of experiments under various settings to evaluate our proposed approach.

5.1 Baseline Comparison Results

In this experiment, we compare the performance of our proposed models with the baseline models, as mentioned in Section 4.4, on the AZ-base dataset. Table 4 reports the evaluation results of the compared models on the five categories of AZ-base. We can observe that GAVI consistently outperforms other baselines across all categories of products. The overall improvement in F_1 score is up to 4.1% compared to OpenBrand. One interesting observation is that both T5 and GAVI outperform the other baselines in terms of F_1 score. The main reason is that both models are generative, meaning that they can identify brand values that are not mentioned in the title. On the other hand, sequence tagging approaches fail to extract such values.

We also notice that, in general, GAVI outperforms the base T5 model in all categories of products. This is mainly because our model is category-aware and is able to learn category-specific embeddings that are more suitable for the identification task. Another key observation is that the performance of GAVI depends on product categories. For

[4] https://pytorch.org/

[5] We only compare our model with the CNN version of OpenBrand as it was shown to have better performance by the authors.

Category	Models	P	R	F_1
Grocery & Gourmet Food	BiLSTM-CRF	74.9	66.0	70.2
	OpenTag	76.0	65.4	70.3
	OpenBrand	77.5	75.4	76.4
	T5	76.5	75.9	76.2
	GAVI	**79.3**	**76.4**	**77.8**
Toys & Games	BiLSTM-CRF	78.9	70.5	74.5
	OpenTag	79.1	70.3	74.5
	OpenBrand	**81.3**	72.0	76.4
	T5	79.7	76.6	78.1
	GAVI	80.3	**77.2**	**78.7**
Sports & Outdoors	BiLSTM-CRF	84.1	75.4	79.5
	OpenTag	84.9	75.0	79.6
	OpenBrand	86.1	77.3	81.5
	T5	82.1	81.5	81.8
	GAVI	**88.1**	**82.3**	**85.1**
Electronics	BiLSTM-CRF	87.8	81.5	84.5
	OpenTag	89.2	79.6	84.2
	OpenBrand	89.7	80.5	84.9
	T5	87.9	81.5	87.8
	GAVI	**90.1**	**88.5**	**89.3**
Automotive	BiLSTM-CRF	90.9	85.0	87.9
	OpenTag	91.6	84.6	87.9
	OpenBrand	**91.8**	85.4	88.5
	T5	90.4	90.5	90.4
	GAVI	91.4	**91.3**	**91.3**

Table 4: Performance comparison between different models on the AZ-base dataset.

Model	P	R	F_1
OpenTag	53.80	33.82	41.53
OpenBrand	55.61	35.46	43.44
T5	67.28	47.90	55.95
GAVI	**70.10**	**53.31**	**60.55**

Table 5: Results on zero-shot brand values.

Category	Models	P	R	F_1
Clothing, Shoes, & Jewelry	BiLSTM-CRF	58.5	42.2	49.0
	OpenTag	60.3	43.5	50.5
	OpenBrand	64.5	45.2	53.2
	T5	64.2	**55.9**	57.4
	GAVI	**64.5**	55.8	**59.8**
Pet Supplies	BiLSTM-CRF	55.0	37.3	44.5
	OpenTag	53.9	38.9	45.2
	OpenBrand	58.2	38.5	46.3
	T5	**64.8**	**51.5**	**57.4**
	GAVI	63.6	49.3	55.6
Cell Phones & Accessories	BiLSTM-CRF	80.1	68.0	73.5
	OpenTag	78.3	67.4	72.4
	OpenBrand	85.2	67.8	75.5
	T5	85.4	81.5	83.4
	GAVI	**85.5**	**81.7**	**83.6**

Table 6: Performance comparison between models on the AZ-new-cat dataset.

example, the gain in recall R in the *Electronics* category (7%) is much higher than the gain in the *Grocery & Gourmet Food* category (1%). By analyzing the errors in the *Grocery & Gourmet Food* category, we discovered that there are certain amount of false negatives in the test set, where the outputs of the model are actually correct, but the labels are wrong. For example, given the following title: "Organo Gold Organic Green Tea (4 Boxes)", the model correctly extracts "Organo Gold" as the brand, but the ground truth is "Organic Green Tea".

5.2 Results of Discovering New Brand Values

We conduct zero-shot extraction experiments to evaluate the generalization performance of our model on unseen brand values. The results on the zero-shot dataset are reported in Table 5. We exclude the BiLSTM-CRF model from this experiment as its not capable to generalize well to new values. It can be seen that our generative models achieve much better results than OpenBrand and OpenTag. For example, the F_1 metric of GAVI significantly increases by 17.1% compared with OpenBrand over all categories in the dataset. This is because generative models use the T5 transformer-based (Vaswani et al., 2017) architecture, which have been shown to outperform the BiLSTM-CRF architecture in zero-shot settings (Wang et al., 2020).

From Table 5, we can also observe that GAVI outperforms the base T5 model on the zero-shot extractions (e.g., by 4.6% F_1 score). This is because knowledge about the category allows the model to exploit similarities across product categories resulting in a better overall performance. However, it is evident that the overall performance of the models is worse as compared to the main results in Table 4. This is expected, as there are no examples from the zero-shot brand values in our training set.

5.3 Results on New Categories

To examine the models ability in generalizing to brand values in new categories, we conduct a set of experiments using the AZ-new-cat benchmark. In these experiments, we train the models on the training set of the AZ-base dataset and evaluate them on three new product categories: *Clothing, shoes, & Jewelry*, *Pet Supplies*, and *Cell Phones & Accessories*. We report the results of our experiments in Table 6. It can be seen that GAVI outperforms all

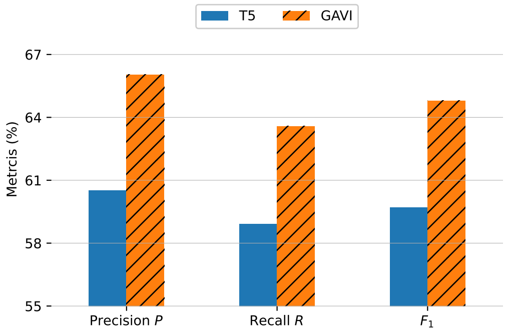


Figure 3: Performance comparison of T5 and GAVI on instances of AZ-base where the brands are not explicitly mentioned in the product description.

the compared baselines. The increase in F_1 score is up to 9.3% as compared to OpenBrand. These results demonstrate the models ability in generalizing to new domains in real-world scenarios.

There are several interesting observations in Table 6. First, the performance of T5 and GAVI is close. This is because using new categories that are not seen during training does not benefit our category-aware implementation. The model is not able to generate category-specific token embeddings at inference time, as they are new categories that are unseen during training. Second, and inline with previous works (Sabeh et al., 2022b), the results on the *Cell Phones & Accessories* category are significantly better than other categories for all compared models. This is because many of the brands in the *Cell Phones & Accessories* category are also present in the *Electronics* category of the training set (e.g., “LG”).

5.4 Results on Implicit Brands Examples

We further conduct a set of experiments on AZ-base to analyze the performance of the models on the instances where the brand value is not explicitly mentioned in the product description. We refer to those examples as *implicit examples* (e.g., the product in Figure 1b). First, we separate the implicit examples in the test set of AZ-base. This resulted in 9k implicit examples. Then, we fine-tune the models on the training set of AZ-base and test them on these implicit examples. Figure 3 shows the evaluation results of T5 and GAVI on the implicit examples. Note that we do not include the other extractive baselines in this experiment as they are not able to extract those implicit brands.

GAVI achieves 64.9% F_1 score on the implicit examples. This indicates the effectiveness of our approach compared to sequence tagging baselines, which are incapable of performing the extraction. In addition, GAVI significantly outperforms the base T5 model in all compared metrics. This clearly indicates that taking the categories into consideration during the generation results in better overall performance.

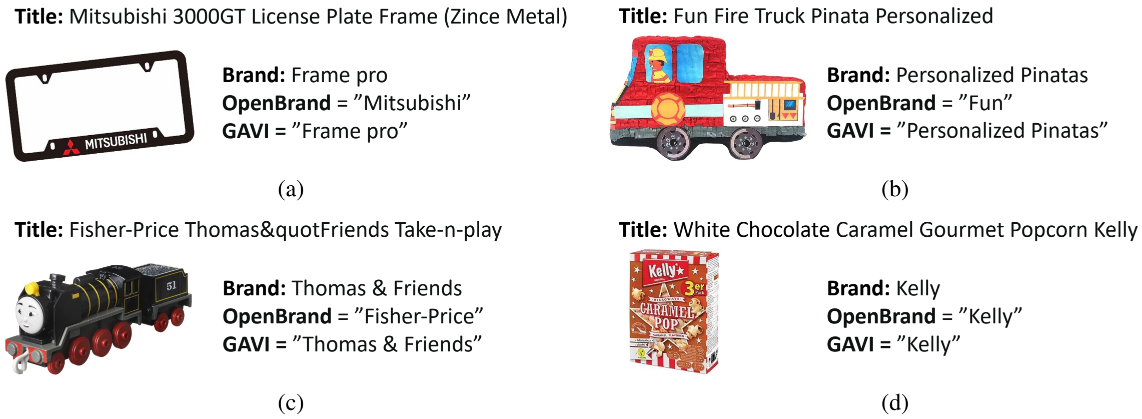
5.5 Examples of Extracted Brand Values

Figure 4 shows examples of product titles and brand values extracted by OpenBrand or GAVI. GAVI is able to identify brands that are not explicitly mentioned in the title: in Figure 4a, “Frame pro” is the valid brand for this product. OpenBrand, which is an extractive sequence tagging model, fails to detect this value. Instead, it extracts “Mitsubishi” as the brand. While GAVI successfully generates “Frame pro” as the correct brand value. In Figure 4b, OpenBrand erroneously extracts “Fun” as the brand for a *Toys & Games* product; on the other hand, GAVI, which considers the product category and textual context, generates the correct brand for this product. Also, in the example of Figure 4c, the model was able to correctly extract the brand value, even though it was mentioned incorrectly in the title.

6 Conclusions and Future Work

Brand value identification is a crucial task in many real-world e-commerce applications. In this work, we propose a novel generative approach for brand value identification. In particular, we employ T5 language model as the backbone of our sequence-to-sequence approach. We infuse category information into the model by highlighting the product categories inside the input. In contrary to previous extractive approaches, our generative method allows us to identify brands beyond the strings mentioned in the product description. Experimental evaluations on public datasets demonstrate the effectiveness of the proposed approach.

We plan to investigate other sequence-to-sequence language models such as BART (Lewis et al., 2020) and GPT (Brown et al., 2020). Also, improving the brand coverage and dealing with the false negatives in the dataset is one of the future directions.


Title: Mitsubishi 3000GT License Plate Frame (Zince Metal)
MITSUBISHI
Brand: Frame pro
OpenBrand = ”Mitsubishi”
GAVI = ”Frame pro”
Title: Fun Fire Truck Pinata Personalized
Brand: Personalized Pinatas
OpenBrand = ”Fun”
GAVI = ”Personalized Pinatas”
Title: Fisher-Price Thomas&quotFriends Take-n-play
Brand: Thomas & Friends
OpenBrand = ”Fisher-Price”
GAVI = ”Thomas & Friends”
Title: White Chocolate Caramel Gourmet Popcorn Kelly
Brand: Kelly
OpenBrand = ”Kelly”
GAVI = ”Kelly”

(a) (b) (c) (d)

Figure 4: Examples of extracted brand values from OpenBrand and GAVI.

Limitations

In this paper, we introduce a novel generative approach for brand value identification from product descriptions. The input to our models is limited to up to around 500 tokens, and the same approach can not be easily applied to longer product descriptions. As far as languages are concerned, the models developed here are English only. To adapt our work to other languages, we need e-commerce datasets to train and evaluate the models in those languages. Also, our models assume that the brand values can always be identified from the context of the product descriptions. We do not consider the case where the context does not include any applicable brand value (i.e., negative values). As future work, we will extend the model and datasets to deal with those negative samples.

References

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Songhao Piao, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML’20. JMLR.org.

Lidong Bing, Tak-Lam Wong, and Wai Lam. 2012. Unsupervised extraction of popular product attributes from web sites. In *Information Retrieval Technology*, pages 437–446, Berlin, Heidelberg. Springer Berlin Heidelberg.

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.

Ying-Hong Chan and Yao-Chung Fan. 2019. A recurrent BERT-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162, Hong Kong, China. Association for Computational Linguistics.

Henrieta Hrablik Chovanová, Aleksander Ivanovich Korshunov, and Dagmar Babčanová. 2015. Impact of brand on consumer behavior. *Procedia Economics and Finance*, 34:615–621. International Scientific Conference: Business Economics and Management (BEM2015).

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

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Rayid Ghani, Katharina Probst, Yan Liu, Marko Krema, and Andrew Fano. 2006. Text mining for product attribute extraction. *SIGKDD Explor. Newsl.*, 8(1):41–48.

Vishrawas Gopalakrishnan, Suresh Parthasarathy Iyengar, Amit Madaan, Rajeev Rastogi, and Srinivasan

Sengamedu. 2012. Matching product titles using web-based enrichment. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, page 605–614, New York, NY, USA. Association for Computing Machinery.

David Ha, Andrew M. Dai, and Quoc V. Le. 2017. Hypernetworks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Touseef Iqbal and Shaima Qureshi. 2022. The survey: Text generation models in deep learning. *Journal of King Saud University - Computer and Information Sciences*, 34(6, Part A):2515–2528.

Giannis Karamanolakis, Jun Ma, and Xin Luna Dong. 2020. TXtract: Taxonomy-aware knowledge extraction for thousands of product categories. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8489–8502, Online. Association for Computational Linguistics.

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

Zornitsa Kozareva, Qi Li, Ke Zhai, and Weiwei Guo. 2016. Recognizing salient entities in shopping queries. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 107–111, Berlin, Germany. 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.

Xiao Ling and Daniel Weld. 2021. Fine-grained entity recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 26(1):94–100.

Ajinkya More. 2016. Attribute extraction from product titles in ecommerce. *CoRR*, abs/1608.04670.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticæ Investigationes*, 30(1):3–26.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. 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 188–197, Hong Kong, China. Association for Computational Linguistics.

Duangmanee Putthividhya and Junling Hu. 2011. Bootstrapped named entity recognition for product attribute extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

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.

Kassem Sabeh, Mouna Kacimi, and Johann Gamper. 2022a. Cave: Correcting attribute values in e-commerce profiles. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, page 4965–4969, New York, NY, USA. Association for Computing Machinery.

Kassem Sabeh, Mouna Kacimi, and Johann Gamper. 2022b. OpenBrand: Open brand value extraction from product descriptions. In *Proceedings of the Fifth Workshop on e-Commerce and NLP (ECNLP 5)*, pages 161–170, Dublin, Ireland. Association for Computational Linguistics.

Umer Shahzad, Salman Ahmad, Kashif Iqbal, Muhammad Nawaz, and Saqib Usman. 2014. Influence of brand name on consumer choice & decision. *IOSR Journal of Business and Management*, 16:72–76.

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

Asahi Ushio, Fernando Alva-Manchego, and Jose Camacho-Collados. 2022. Generative language models for paragraph-level question generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 670–688, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Damir Vandic, Jan-Willem van Dam, and Flavius Frasincar. 2012. Faceted product search powered by the semantic web. *Decision Support Systems*, 53(3):425–437.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Qifan Wang, Li Yang, Bhargav Kanagal, Sumit Sanghai, D. Sivakumar, Bin Shu, Zac Yu, and Jon Elsas. 2020. Learning to extract attribute value from product via question answering: A multi-task approach. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 47–55. ACM.

Dongling Xiao, Han Zhang, Yukun Li, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. Ernie-gen: An enhanced multi-flow pre-training and fine-tuning framework for natural language generation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20.

Huimin Xu, Wenting Wang, Xin Mao, Xinyu Jiang, and Man Lan. 2019. Scaling up open tagging from tens to thousands: Comprehension empowered attribute value extraction from product title. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5214–5223, Florence, Italy. Association for Computational Linguistics.

Jun Yan, Nasser Zalmout, Yan Liang, Christan Grant, Xiang Ren, and Xin Luna Dong. 2021. AdaTag: Multi-attribute value extraction from product profiles with adaptive decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4694–4705, Online. Association for Computational Linguistics.

Li Yang, Qifan Wang, Zac Yu, Anand Kulkarni, Sumit Sanghai, Bin Shu, Jon Elsas, and Bhargav Kanagal. 2022. MAVE: A product dataset for multi-source attribute value extraction. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, pages 1256–1265. ACM.

Yi Zhang et al. 2015. The impact of brand image on consumer behavior: A literature review. *Open journal of business and management*, 3(01):58.

Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. Opentag: Open attribute value extraction from product profiles. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 1049–1058. ACM.

Simple, Simpler and Beyond: A Fine-Tuning BERT-Based Approach to Enhance Sentence Complexity Assessment for Text Simplification

Lucía Ormaechea[1,2], **Nikos Tsourakis**[1], **Didier Schwab**[2],
Pierrette Bouillon[1] and **Benjamin Lecouteux**[2]
[1] TIM/FTI, University of Geneva, 40 Boulevard du Pont-d'Arve – Geneva, Switzerland
{firstName.lastName}@unige.ch
[2] Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG – Grenoble, France
{firstName.lastName}@univ-grenoble-alpes.fr

Abstract

Automatic text simplification models face the challenge of generating outputs that, while being indeed simpler, still retain some complexity. This stems from the inherently relative nature of simplification, wherein a given text is transformed into a relatively simpler version, which does not necessarily equate to simple. We thus aim to propose a finer-grained method to assess sentence complexity in French. Our solution comprises three models, in which two address absolute and relative sentence complexity assessment, while the third focuses on measuring simplicity gain. By employing this triad of models, we aim to offer a comprehensive approach to qualify and quantify sentence simplicity. Our approach utilizes FlauBERT, fine-tuned for classification and regression tasks. Based on our three-dimensional complexity analysis, we provide the WiViCo dataset, comprising 46,525 aligned *complex-simpler* pairs, which can be further leveraged to fine-tune large language models to automatically generate simplified texts, or to assess text complexity with greater granularity.

1 Introduction

Automatic Text Simplification (ATS) aims at producing a simpler version of a given input text, while still preserving its original information, semantic coherence and grammaticality (Horn et al., 2014). The resulting text is expected to be linguistically less complex, which can in turn have an interest from a *human-oriented perspective*, so as to provide with adapted texts for different target readers, like children (De Belder and Moens, 2010) or people with dyslexia (Rello et al., 2013); and a *machine-oriented perspective*, as a pre-processing step for other NLP applications like information extraction (Evans and Orasan, 2019).

Nevertheless, ATS models are subject to generating outputs that, while being indeed simpler, still retain a level of complexity. This arises from the inherently relative nature of simplification, in which a given reference text is rewritten into a comparatively simpler version. Yet, simpler does not necessarily equate to simple, and can result in outputs that still exhibit complex linguistic features.

Predicting sentence complexity seems a valuable ancillary task in this respect, as it can help evaluate the simplification effectiveness of the generated output. In addition, it can contribute to the automatic creation of monolingual *complex-simpler* pairs, which are a scarce resource in ATS, especially for less resource-rich languages than English. Prior research has often addressed sentence complexity assessment by relying on binary classification models (Paetzold and Specia, 2016; Stajner et al., 2017), through which an input is categorized as either *complex* or *simple* on an absolute basis. However, this approach proves somewhat coarse in the context of simplification, considering its acknowledged relative nature. Since ATS models operate based on a provided text, we believe that estimating the sentential complexity should also be conducted in a reference-aware manner.

In this paper, we aim to contribute with a BERT-based finer-grained method to assess sentence complexity, specifically in French. Despite its substantial resources, ATS research on this language remains largely unexplored given the scarcity of parallel simplification data. To alleviate this issue, we introduce a new triad of increasingly fine-grained models so as to: *i*) determine whether a sentence is inherently *complex* or *simple*; *ii*) assess if the second sentence in a pair is simpler than the first; and *iii*) measure the simplification gain achieved by the second sentence in comparison to the original one. Additionally, based on the proposed method, we provide a general-purpose parallel sentence simplification dataset for French language[1].

[1] Which is publicly released on the following GitHub repository: https://github.com/lormaechea/wivico.

2 Background and related work

2.1 Simple and simpler: a fundamental distinction often omitted in ATS

The performance of ATS models is normally judged upon three criteria (Martin, 2021): *i*) how *fluent* the simplified output is; *ii*) how well the *meaning* of source text is preserved in the output; and most notably, *iii*) how simple it is compared to the original unsimplified text. A successful model is thus expected to produce a fluent, lossless-meaning text that is comparatively simpler in form than its original counterpart. This implies that the system is not necessarily designed to generate *simple text*, but rather to achieve or satisfice a simplicity gain with respect to a given text. In other words, the model is aimed at producing a comparatively simpler version of a text, according to a provided input. Yet *simpler* does not equal *simple* by definition. A complex text can be transformed into a relatively simpler version, but still show complex features that would make them inadequate to the constraints of simple language.

Then the question that arises is: what is the notion of *simple*? Is there such a thing as an absolute and objective simplicity that defines one particular text? The concept of *simple language* has been extensively investigated in prior literature, especially in the context of text accessibility. It has been broadly defined as a variety of language that shows low lexical and syntactic complexity (Klaper et al., 2013). Nevertheless, providing proper simplified texts requires a more precise delineation, as it is greatly influenced by the needs of specific target readers (*e.g.*, individuals with cognitive disabilities, foreign language learners, children, *etc*.), which condition the preferred simplification operations accordingly. As can be noted, the audience is not a negligible factor, as it shows that text simplification is a strongly subject-dependent task: the perception of a text as being more easily accessible or comprehensible may vary substantially according to the target reader (Dmitrieva et al., 2021).

In recent years, the growing awareness of the eventual reading comprehension difficulty arisen by some types of documents (*e.g.*, technical, administrative, but also general-domain) (Stajner, 2021), as well as the regulations ratified from institutional frameworks (Nomura et al., 2010), has fostered the definition of easy-to-understand manual simplification style guides, such as *Easy Language* or *Plain Language* (Maaß, 2020). These initiatives were created to provide standards for the writing of comprehensibility-enhanced texts, and to guarantee the quality and appropriateness of the resulting simplifications. Nonetheless, such guidelines often advise the use of overly broad or imprecise simplification-oriented rules, such as the usage of short sentences and simple words, or the avoidance of non-essential information (Candido et al., 2009). Such haziness hinders their eventual applicability within automated text simplification solutions. And, more importantly, it makes it difficult to objectively quantify the extent to which a text complies with a specific guideline (Fajardo et al., 2013; Sutherland and Isherwood, 2016), thus obfuscating a consensual definition of *simple language* and a common characterization of *simple text*.

2.2 Existing approaches for building parallel text simplification corpora

The creation of relevant resources for text simplification is a crucial procedure for the subsequent training and evaluation of data-driven ATS models. However, it poses a significant challenge due to the intricacies associated with defining *simplicity*, as discussed earlier, and also the strong reliance on monolingual parallel corpora comprising representative simplified texts and their corresponding complex references. The paucity of such data collections has significantly hindered progress on this task, both method- and language-wise. To mitigate this issue, previous research has employed two approaches for building parallel *complex-simple(r)* text resources: *manual* and *automatic*, with a special focus on sentence-level simplifications.

Manually-created Manually crafted monolingual parallel corpora for ATS are usually created from scratch, by asking experts (*i.e.*, teachers, translators or speech therapists) to simplify a set of texts (usually genre- or domain-specific), for a particular audience (Brunato et al., 2022). By relying on pre-existing or *ad hoc* target-aware style guidelines, and professional editors' expertise, the resulting sentence simplification pairs are expected to provide a reliable and high-quality parallel dataset.

On this basis, several datasets have been released, such as NEWSELA (Xu et al., 2015), in English and Spanish, PORSIMPLES (Aluisio and Gasperin, 2010) in Brazilian Portuguese, or ALECTOR (Gala et al., 2020) in French. Parallel corpora derived from this approach are notable for their highly reliable simplification operations performed on the original text. However, this process is costly, both economically and time-wise, due to the requirement of trained human editors. Furthermore, it has an impact on the reduced size of the resulting dataset, which with the exception of NEWSELA, does not easily support the implementation of ML algorithms that are able to infer the transformations to generate simplified text.

Automatically-created With the goal of providing with ATS-oriented high-scale parallel monolingual datasets, automatic data acquisition approaches rely on existing comparable corpora (usually Wiki-based) that associate standard texts with their simplified versions. These resources are later used to extract *complex-simple(r)* sentence pairs, giving rise to labeled data collections, like WIKISMALL (Zhu et al., 2010), EW-SEW (Hwang et al., 2015) or WIKILARGE (Zhang and Lapata, 2017).

While being widely used in the training of ATS models in prior literature (Nisioi et al., 2017; Martin et al., 2020; Sheang and Saggion, 2021), the adequacy of the simplifications within these datasets has been called into question (Xu et al., 2015). This is due to the eventual disparity between the source text and its comparatively simpler counterpart, given the fact that comparable corpora being used are often written independently. In addition to this, their limited controllability has also been debated, since it appears difficult to determine to what extent they observe any style manual, or whether the performed simplifications are target-aware or target-oblivious. Nor is it any less of an impediment that such resources are often solely existing in English, leading data-driven ATS in less resource-rich languages to be harder to implement.

Yet, the main reason to emphasize the unsuitability of these datasets is based on the eventual suboptimality of the methods used to mine register-diversified comparable corpora. So as to capture monolingual parallel data that is relevant for ATS, prior research has typically relied on automatic alignment algorithms and semantic similarity scores (Paetzold et al., 2017; Stajner et al., 2018; Nikolov and Hahnloser, 2019; Sun et al., 2023). Although these strategies are prone to error, they aid in assessing the semantic closeness between two sentences, and thus serve as a proxy for meaning preservation. However, they do not suffice on their own, as they fail to ascertain whether the target text genuinely constitutes a simpler version with respect to the corresponding input. Given that simplicity gain is a *sine qua non* condition for a simplified text to be considered valid, recent studies have explored the use of classification and regression models to estimate sentence complexity, as we will see below.

2.3 Automatic assessment of sentence complexity

Automatically determining the complexity of a sentence proves to be a valuable ancillary task for ATS, as it can potentially serve as a preliminary step in creating labeled simplification data. Additionally, it can aid in evaluating the simplification effectiveness of the generated output.

Prior literature has approached sentence complexity prediction in various ways, depending on the ultimate objective. This typically includes: *i*) detecting the complex sentences needing to be simplified, and *ii*) quantifying the degree of simplification achieved within a pair. As a result, it has had an impact on the approach used for such assessment. So as to address the first goal, previous works have mainly employed absolute complexity classifiers. These models assign a discrete label to an input text that represents its difficulty. This can in turn be treated as a binary classification problem (Paetzold and Specia, 2016; Stajner et al., 2017) or a multi-class discrimination problem, if a greater granularity is considered (Vajjala and Meurers, 2014; Khallaf and Sharoff, 2021). On the other side, relative sentence complexity classifiers (Ambati et al., 2016) and, more particularly, regression models have been prioritized to address the second objective (Iavarone et al., 2021), as they can represent linguistic complexity in a continuum, and help predict the degree of complexity reduction obtained by a simplified sentence.

It is also worth noting that such regressors have commonly been used from the perspective of automatic readability assessment (Lee and Vajjala, 2022). While it is a complementary notion to that of simplification, they are not equivalent concepts. Readability primarily focuses on language clarity and accessibility, and it does not strictly target the *meaning preservation* and *simplicity gain* relation. In addition to this, readability formulae were designed for a document-level application, which means that they may not be completely reliable on a sentential-level (Stajner et al., 2017). This suggests the need to introduce new metrics within ATS, so as to properly quantify the gain or loss of simplicity in a *complex-simpler* pair.

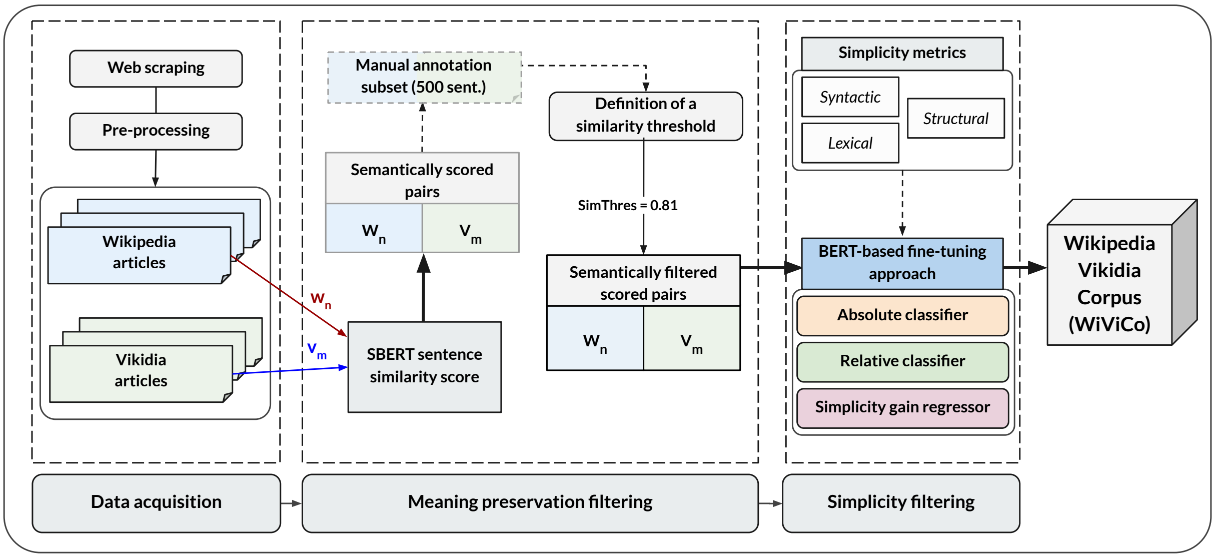

Web scraping
Pre-processing
Wikipedia articles
Vikidia articles
Manual annotation subset (500 sent.)
Semantically scored pairs
W_n
V_m
SBERT sentence similarity score
Definition of a similarity threshold
SimThres = 0.81
Semantically filtered scored pairs
Simplicity metrics
Syntactic
Structural
Lexical
BERT-based fine-tuning approach
Absolute classifier
Relative classifier
Simplicity gain regressor
Wikipedia Vikidia Corpus (WiViCo)
Data acquisition
Meaning preservation filtering
Simplicity filtering

Figure 1: Overview of the pipeline to obtain *complex-simpler* sentence pairs from the French Wikipedia and Vikidia.

3 Corpora

As previously stated, automatically determining the complexity of a sentence (or a pair of sentences) can potentially serve as a helpful preliminary step in creating labeled simplification data in languages such as French, where ATS-specific aligned data is scarce. In this section, we showcase the corpora we used to make such prediction as well as to automatically mine *complex-simpler* pairs.

3.1 WikiLarge-Fr

Assessing sentence simplicity in an automatic manner is generally based on data-driven approaches. Considering this, we opted to rely on WikiLarge (Zhang and Lapata, 2017), a well-established dataset that has been utilized to develop and refine simplification models in previous ATS research. However, a significant obstacle was encountered since the texts in WikiLarge were originally written in English, requiring to be translated into French. To tackle this issue, we employed Google Translate to obtain the respective translations for every pair and produced WikiLarge-Fr.

WikiLarge-Fr	
Train size	105,420
Dev size	13,177
Test size	13,179
Total	131,776

Table 1: Overview of size (in sentence pairs) and data distribution of the WikiLarge-Fr dataset.

We identified that certain pairs were too similar during this process, so we kept those with a Levenshtein distance of less than 0.95. We then split the data into a train, validation, and test set using an 80:10:10 split and stratification (see Table 1).

3.2 Wikipedia-Vikidia data compilation

Prior studies have highlighted the potential use of Wiki-based articles for the creation of ATS resources (Brouwers et al., 2012). For this reason, we decided to use the French-language editions of register-differentiated comparable corpora to subsequently extract parallel simplification pairs. More precisely, we relied on Wikipedia and Vikidia, where the latter constitutes an adapted version of the former, and was created to provide with texts that can be more easily understandable by children between 8 and 13 years old. At present, French Vikidia comprises about 40*k* articles, which makes it a significant resource for ATS. Notwithstanding French is a reasonably well-resourced natural language, the available aligned data for this task is limited (Seretan, 2012; Cardon and Grabar, 2019).

In order to retrieve the textual content from the articles of both sources, we extracted the complete URL list of articles from Vikidia using the web scraping pipeline described in Ormaechea and Tsourakis (2023). The output yielded a total of 34,357 article links[2]. We later parsed the HTML content to find the corresponding Wikipedia articles, by relying on inter-language links. Afterwards, we tokenized the text content and segmented

[2] As of April 14th, 2023.

it into sentences. We finally filtered out the sentences exceeding 128 word pieces, so as to avoid an eventual truncation when encoded into a sentence embedding.

4 Meaning preservation filtering

As discussed in Section 2.1, the output produced by an ATS model is expected to meet two primary conditions: *i*) retain the meaning and information conveyed in the input text, and *ii*) obtain a linguistic simplicity gain with respect to the reference. Based on this definition, we addressed these two dimensions sequentially. In order to determine suitable *complex-simpler* pairs for ATS, we must first assess whether they are semantically equivalent[3].

We thus implemented a meaning preservation filtering method to identify the Wiki-Viki pairs exhibiting a high semantic overlap. To this effect, we relied on SBERT (Reimers and Gurevych, 2019), which modifies the pretrained BERT network (Devlin et al., 2019) by using a siamese architecture to compute sentence embeddings[4]. After mapping the sentences to a 768-dimensional dense vector space, we computed the cosine similarity for the resulting encoded pairs.

Once such values were obtained, we needed to assess which pairs showed sufficient semantic consistency. To this end, we chose to rely on a manual annotation of 500 randomly picked sentence pairs from our initial dataset. Two subjects were selected for this purpose. They were given three judgment labels to conduct the annotation: *valid*, where the meaning from source to target is fully preserved; *partially valid*, where information is partially lost from source to target or vice versa; and *non-valid*, where information between the two sentences diverges. After the first annotation round, the two experts convened to discuss and reached a consensus, resulting in a Cohen's kappa score of 0.87. With 500 annotated sentence pairs at our disposal, we plotted the distribution of the SBERT scores for each judgment label. On average, *valid* pairs show higher SBERT-derived values, which confirms a direct correlation between SBERT scoring and human judgments on sentence similarity. The mean score for *valid* pairs was 0.81, which we consider the cutoff threshold for the semantic filtering step.

5 Simplicity filtering

After addressing the meaning preservation dimension, we focused on how to extract the simplicity gain obtained by the target sentence with respect to the reference. Our approach consists of three distinct steps to assess absolute and relative simplification and estimate a gain score (as shown in Figure 1), and aims to properly address the relative nature of *simplification*. An absolute binary categorization of a sentence as *complex* or *simple* seems somewhat insufficient and not suited for ATS. Indeed, a complex sentence (C) being transformed into a simple one (S) results in a *simplification*. Conversely, a S→C process gives rise to a *complexification*. Nevertheless, an absolute classifier can equally categorize a source and target sentences as C→C or S→S. Given that *simplification* and *complexification* operations are reference-dependent, they may validly occur in both cases.

Because there are several phenomena involved within simplicity assessment, we split the problem into an increasingly fine-grained approach. First, we incorporated the WIKILARGE-FR dataset to elicit pairs of *complex-simpler* sentences that can be used to fine-tune different versions of FlauBERT (Le et al., 2020). For the classification task, we created two models: one to assess the simplicity of each sentence in the pair, and another to determine whether the target sentence is simpler than the corresponding source. Subsequently, based on a set of features, we calculated the simplicity gain for each pair that allowed the creation of a regressor model to automate this process. For a clearer depiction of the specific steps involved, refer to Figure 2.

5.1 Classification models for sentence complexity

Fine-tuning pre-trained classification models can help leverage their learned knowledge and transfer it to a new classification task. By adapting the model to the target task with labeled data, we can improve its generalization, capture domain-specific nuances, and achieve better results. In our work, we incorporated a specific architecture based on the FlauBERT language model to perform sentence complexity classification. It is a variant of the model that has been adapted specifically for sequence classification. In this architecture, the model is combined with additional layers and a classification head to enable it to classify sequences into different categories.

[3] If their meaning is divergent, no assessment on simplicity gain is applicable.

[4] We used multilingual sentence transformers: `https://huggingface.co/sentence-transformers/paraphrase-xlm-r-multilingual-v1`.

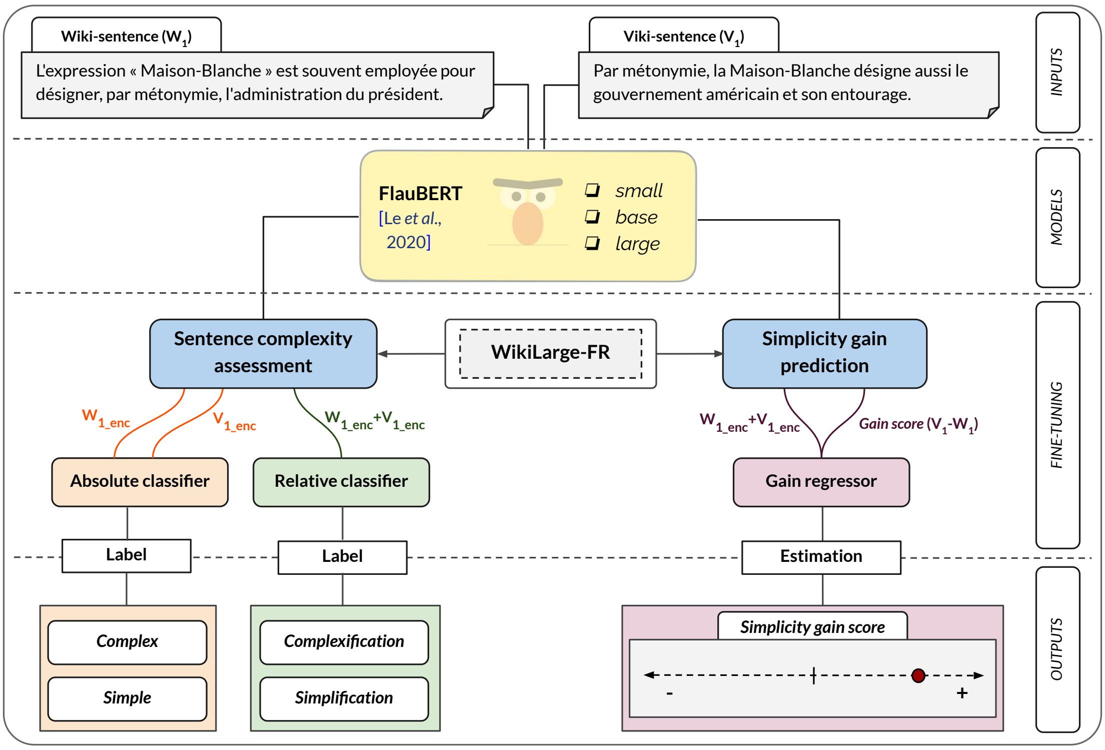


Figure 2: Overview of the simplicity assessment task.

5.1.1 Absolute sentence complexity assessment

In the first experiment, we treated each sentence in the input pairs independently to determine whether it is categorized as *simple* or *complex*. To achieve this, we assigned a binary label for each of the sentences in the WIKILARGE-FR dataset (see Section 3.1). The performance on the test set is presented on the left side of Table 2. Utilizing different variants of the FlauBERT model, we contrasted the performance between each baseline model (*untuned*) and the one after training (*tuned*). We observe significant improvement in the second case, which is similar to all three variants. The baseline untuned models' performance was no better than random chance in distinguishing between the two classes (∼50%) versus the tuned ones (∼70%). It is worth noting that the small version of the untuned FlauBERT model is partially trained, which may impact its performance. Nevertheless, it was included for debugging purposes.

5.1.2 Relative sentence complexity assessment

The second classifier aims to assess the relative simplification between the source and target sentence pairs, answering the question of whether the second is a *simpler* version of the first. To accomplish this, we juxtaposed the sentences alternating their order into two sets of pairs to signify either *simplification* or *complexification*. This time, we significantly improved the baseline performance (∼50% versus ∼93%). To reinforce the validity of the previous outcome, we also utilized the manually annotated dataset of Section 4, which included human annotations of relative simplification. The results shown on the right side of Table 2 corroborate our previous assessment. As the dataset is imbalanced, the baseline classifiers' performance mirrors the class distribution and can largely be attributed to chance. However, the tuned models improve those significantly (∼94%).

5.2 A regression model for simplicity gain

The classification models presented above allow us to discern in a binary manner whether a sentence is *complex* or *simple*, or whether a pair of sentences has undergone a process of *simplification* or *complexification*. However, these models lack the capacity to indicate to what extent a target sentence is *simpler* than its original counterpart. For these reasons, we have aimed to quantify the simplification shift produced within a pair of classically categorized *complex-simple* sentences, with the training of a regression model. In this way, we have sought to measure the *simplicity gain* achieved from the original sentence to its simplified version.

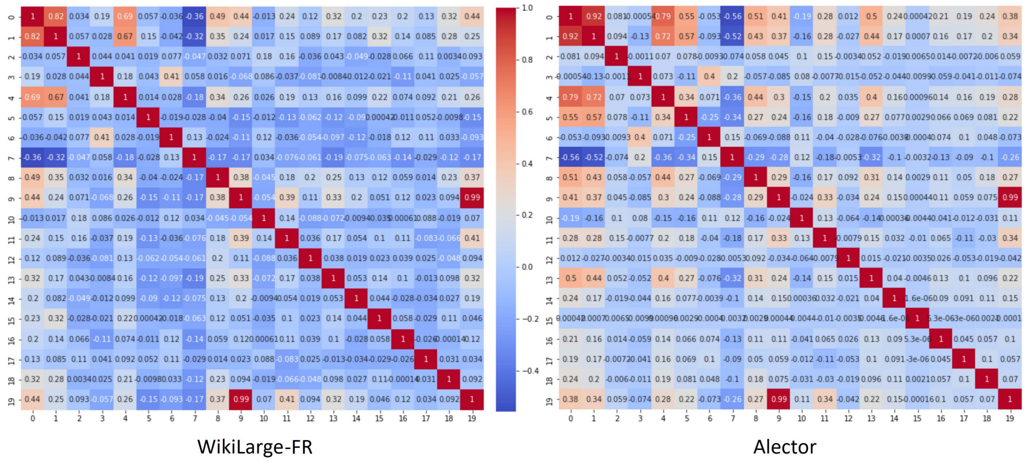


Figure 3: Correlation heatmaps among the feature gains for the WIKILARGE-FR and ALECTOR datasets.

Classification task	AC		RC			
Evaluation dataset	Test set		Test set		Manual set	
Transformer model	untuned	tuned	untuned	tuned	untuned	tuned
`flaubert-small`	49.54	70.11	49.78	92.99	34.58	92.52
`flaubert-base`	50.97	69.82	49.88	93.82	36.45	93.46
`flaubert-large`	52.29	69.19	52.18	94.16	75.71	95.33

Table 2: Accuracy results in % obtained for the *absolute complexity classifier* (AC) on the test set, and for the *relative complexity classifier* (RC) on the test and manual evaluation sets.

As noted in Section 2.3, similar regression models have been used from a readability perspective, but they prioritize the measurement of clarity and accessibility aspects, and do not explicitly address the challenges of ATS. This is why we sought to examine the quantification of the simplicity gain.

5.2.1 Definition of features

We extracted a set of pertinent features, shown in Table 4, that were chosen on the basis of previous literature regarding sentence simplicity assessment (Tanguy and Tulechki, 2009; Brunato et al., 2022). These describe the WIKILARGE-FR dataset along three dimensions and are grouped into structural, lexical, and syntactic groups. Based on these features, we calculated their values for each sentence in the pair and performed an element-wise subtraction. The result is a list containing the differences between the elements in the same positions of the original feature lists that we also standardized.

While using a predictive model to estimate the *simplicity gain* from *complex-simpler* pairs might not be necessary when a direct calculation process is available, there are potential benefits to consider. Predictive models can assist in quality assessment by identifying cases where direct calculations may falter due to assumptions or heuristics. They offer generalization capabilities, making predictions for new data and variations that the direct process may not cover. Additionally, these models can uncover hidden patterns, adapt to changes in data distributions, and provide robustness against noisy or imperfect data, enhancing their value in real-world scenarios. For that reason, LLMs can be beneficial by leveraging their capacity to comprehend and learn from intricate language patterns in the data.

To tackle the challenge of collinearity, we calculated the correlation of the simplicity gains shown in the left heatmap of Figure 3. This heatmap aids in detecting patterns and dependencies among the features. This helps to identify the impact of each one on the overall simplicity gain and to decide on which to keep in the subsequent analysis. We observe that certain pairs demonstrate a high correlation, like *Sentence length* and *Number of words* (row: 0 – col: 1) or *IDT* and *IDT-DLT* (row: 9 – col: 19). We therefore excluded the second feature in each pair, ending with 18 features in total.

We also performed a symmetric analysis on the aforementioned ALECTOR dataset (shown in the right heatmap of Figure 3). Given that it was manually created by expert linguists, the produced simplifications are expected to be highly reliable. This in turn helps to reinforce our decision to maintain or exclude features according to their relevance to the simplicity assessment. Interestingly, we observe similar patterns of correlation, indicating that the features have a similar effect in both datasets.

5.2.2 Simplicity gain estimation

Similarly to the classification tasks, we fine-tuned FlauBERT for regression. By utilizing the Mean Squared Error (MSE) as the loss function, Adam optimizer and a batch size of 16, we trained FlauBERT to learn to map its linguistic representations to continuous target variables. The input received by the regressor consisted on the *complex-simpler* pairs appended to their simplicity gain score, with a maximum input size of 512 tokens.

GR		
Evaluation dataset	Test set	
Transformer model	untuned	tuned
`flaubert-small`	1.89	0.39
`flaubert-base`	1.18	0.35
`flaubert-large`	4.59	0.23

Table 3: MSE scores from the *gain regressor* (GR).

Table 3 contrasts the performance on the test set using either an untuned or a tuned FlauBERT model. We observe a significant improvement in all three cases. Specifically, the tuned models achieved a much lower MSE, demonstrating their ability to capture underlying patterns in the data and provide more accurate predictions. The `flaubert-large` model yields the best performance with an MSE equal to 0.23, which seems still insufficient in the context of our application. These results may suggest further exploration in the optimization of the model hyperparameters, but they may also point towards a broader categorization of each pair based on a range of gain values.

5.3 Wikipedia-Vikidia Corpus (WiViCo)

Having this triad of models in place, we were able to finally implement our fine-grained method on sentence simplicity to extract relevant pairs for ATS. To do so, we implemented our best performing models on the compiled data introduced in Section 3.2. As a result, we were able to generate the Wikipedia-Vikidia Corpus (WiViCo), that contains 46,525 aligned sentence pairs[5]. These include standard C→S labeled examples, but also C→C and S→S ones, where a simplification operation was performed (as can be seen in Appendix B).

6 Conclusions and further work

This paper presents an increasingly fine-grained approach for assessing sentence simplicity. Through a comprehensive three-dimensional analysis, our objective was to estimate sentence simplicity in a manner suitable for ATS, which is an inherently relative operation. Additionally, we believe that our work can serve as a relevant and reproducible method to automatically create parallel simplification datasets. This can in turn be of great interest for reasonably well-resourced natural languages like French that still lack sufficient resources for the ATS task. Consequently, we provide public access to the dataset that derives from the application of our approach, WiViCo. This may allow other researchers interested in this field to further use this resource to fine-tune LLMs for the task at hand, or to assess text complexity in a finer-grained manner.

As for the limitations of this work, it is important to note that due to the volume of the WikiLarge corpus, we had to resort to Google Translate to obtain the corresponding French texts, without manually assessing the correctness of the produced outputs. A possible workaround to this drawback would be to compare a subset of the produced WikiLarge-Fr with its original counterpart and conduct a human evaluation of translation quality.

On another note, an extension of our investigations points to the creation of configurable ATS models. We could incorporate our triad of models into a larger pipeline designed for text simplification and use them to rank a set of candidate simplified sentences, with the goal of selecting the most simplified sentence that best preserves the original meaning of the input. Similarly, the fine-tuned model can serve as a guide during the simplification process by providing a continuous feedback signal to a generative ATS model and therefore adjust its output to attain a desired level of simplification.

Last but not least, we also intend to work on improving the interpretability of the assigned score for simplicity gain. While based on a calculation resulting from established linguistic features for text simplicity, we believe it is also necessary to contrast such scores to human judgments. By doing so, we can examine the correlation between the two in more depth, and measure the significance of each feature in the simplicity gain estimation.

[5] Appendix C provides a detailed description of the dataset).

Acknowledgements

This work is part of the PROPICTO (French acronym standing for *PRojection du langage Oral vers des unités PICTOgraphiques*) project, funded by the Swiss National Science Foundation (N°197864) and the French National Research Agency (ANR-20-CE93-0005).

References

Sandra Aluisio and Caroline Gasperin. 2010. Fostering Digital Inclusion and Accessibility: The PorSimples project for Simplification of Portuguese Texts. In *Proceedings of the NAACL HLT Young Investigators Workshop on Computational Approaches to Languages of the Americas*, pages 46–53. Association for Computational Linguistics.

Bharat Ram Ambati, Siva Reddy, and Mark Steedman. 2016. Assessing Relative Sentence Complexity using an Incremental CCG Parser. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1051–1057. Association for Computational Linguistics.

Laetitia Brouwers, Delphine Bernhard, Anne-Laure Ligozat, and Thomas François. 2012. Simplification Syntaxique de Phrases pour le Français. In *Actes de la Conférence Conjointe JEP-TALN-RECITAL*, pages 211–224.

Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi. 2022. Linguistically-Based Comparison of Different Approaches to Building Corpora for Text Simplification: A Case Study on Italian. *Frontiers in Psychology*, 13.

Arnaldo Candido, Erick Maziero, Lucia Specia, Caroline Gasperin, Thiago Pardo, and Sandra Aluisio. 2009. Supporting the Adaptation of Texts for Poor Literacy Readers: A Text Simplification Editor for Brazilian Portuguese. In *NAACL HLT Workshop on Innovative Use of NLP for Building Educational Applications*, pages 34–42.

Rémi Cardon and Natalia Grabar. 2019. Parallel Sentence Retrieval From Comparable Corpora for Biomedical Text Simplification. In *Proceedings - Natural Language Processing in a Deep Learning World*, pages 168–177.

Jan De Belder and Marie-Francine Moens. 2010. Text Simplification for Children. In *Workshop on Accessible Search Systems*, pages 19–26.

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

Anna Dmitrieva, Antonina Laposhina, and Maria Lebedeva. 2021. A Comparative Study of Educational Texts for Native, Foreign, and Bilingual Young Speakers of Russian: Are Simplified Texts Equally Simple? *Frontiers in Psychology*, 12.

Richard Evans and Constantin Orasan. 2019. Sentence Simplification for Semantic Role Labelling and Information Extraction. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 285–294.

Inmaculada Fajardo, Vicenta Clemente, Antonio Ferrer, Gema Tavares, Marcos Gómez, and Ana Hernández. 2013. Easy-to-read Texts for Students with Intellectual Disability: Linguistic Factors Affecting Comprehension. *Journal of Applied Research in Intellectual Disabilities (JARID)*, 27:212–225.

Núria Gala, Anaïs Tack, Ludivine Javourey-Drevet, Thomas François, and Johannes C. Ziegler. 2020. Alector: A Parallel Corpus of Simplified French Texts with Alignments of Misreadings by Poor and Dyslexic Readers. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1353–1361.

Colby Horn, Cathryn Manduca, and David Kauchak. 2014. Learning a Lexical Simplifier Using Wikipedia. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 458–463.

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

Benedetta Iavarone, Dominique Brunato, and Felice Dell'Orletta. 2021. Sentence Complexity in Context. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 186–199. Association for Computational Linguistics.

Nouran Khallaf and Serge Sharoff. 2021. Automatic Difficulty Classification of Arabic Sentences. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 105–114. Association for Computational Linguistics.

David Klaper, Sarah Ebling, and Martin Volk. 2013. Building a German/Simple German Parallel Corpus for Automatic Text Simplification. In *Proceedings of the Second Workshop on Predicting and Improving Text Readability for Target Reader Populations*, pages 11–19. Association for Computational Linguistics.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbé, Laurent Besacier, and Didier

Schwab. 2020. FlauBERT: Unsupervised Language Model Pre-training for French. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*. European Language Resources Association.

Justin Lee and Sowmya Vajjala. 2022. A Neural Pairwise Ranking Model for Readability Assessment. In *Findings of the Association for Computational Linguistics*, pages 3802–3813. Association for Computational Linguistics.

Christiane Maaß. 2020. *Easy Language – Plain Language – Easy Language Plus: Balancing Comprehensibility and Acceptability*. Frank & Timme.

Louis Martin. 2021. *Automatic Sentence Simplification using Controllable and Unsupervised Methods*. Ph.D. Thesis, Sorbonne Université.

Louis Martin, Éric de la Clergerie, Benoît Sagot, and Antoine Bordes. 2020. Controllable Sentence Simplification. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4689–4698.

Nikola I. Nikolov and Richard Hahnloser. 2019. Large-Scale Hierarchical Alignment for Data-driven Text Rewriting. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 844–853.

Sergiu Nisioi, Sanja Stajner, Simone Paolo Ponzetto, and Liviu P. Dinu. 2017. Exploring Neural Text Simplification Models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 85–91.

Misako Nomura, Gyda Skat Nielsen, International Federation of Library Associations and Institutions, and Library Services to People with Special Needs Section. 2010. *Guidelines for Easy-to-Read Materials*. IFLA Headquarters.

Lucía Ormaechea and Nikos Tsourakis. 2023. Extracting Sentence Simplification Pairs from French Comparable Corpora Using a Two-Step Filtering Method. In *Proceedings of the 8th Swiss Text Analytics Conference 2023*. Association for Computational Linguistics.

Gustavo Paetzold, Fernando Alva-Manchego, and Lucia Specia. 2017. MASSAlign: Alignment and Annotation of Comparable Documents. In *Proceedings of the IJCNLP, System Demonstrations*, pages 1–4.

Gustavo Paetzold and Lucia Specia. 2016. SemEval 2016 Task 11: Complex Word Identification. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, pages 560–569. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3982–3992. Association for Computational Linguistics.

Luz Rello, Ricardo Baeza-Yates, and Horacio Saggion. 2013. DysWebxia: Textos Más Accesibles Para Personas con Dislexia. *Procesamiento del Lenguaje Natural*, 51.

Violeta Seretan. 2012. Acquisition of Syntactic Simplification Rules for French. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC)*, pages 4019–4026.

Kim Cheng Sheang and Horacio Saggion. 2021. Controllable Sentence Simplification with a Unified Text-to-Text Transfer Transformer. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 341–352. Association for Computational Linguistics.

Sanja Stajner. 2021. Automatic Text Simplification for Social Good: Progress and Challenges. In *Findings of the Association for Computational Linguistics*, pages 2637–2652. Association for Computational Linguistics.

Sanja Stajner, Marc Franco-Salvador, Paolo Rosso, and Simone Paolo Ponzetto. 2018. CATS: A Tool for Customized Alignment of Text Simplification Corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*, pages 3895–3903.

Sanja Stajner, Simone Paolo Ponzetto, and Heiner Stuckenschmidt. 2017. Automatic Assessment of Absolute Sentence Complexity. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI*, pages 4096–4102.

Renliang Sun, Zhixian Yang, and Xiaojun Wan. 2023. Exploiting Summarization Data to Help Text Simplification. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 39–51. Association for Computational Linguistics.

Rebekah Sutherland and Tom Isherwood. 2016. The Evidence for Easy-Read for People With Intellectual Disabilities: A Systematic Literature Review: The Evidence for Easy-Read for People With Intellectual Disabilities. *Journal of Policy and Practice in Intellectual Disabilities*, 13:297–310.

Ludovic Tanguy and Nikola Tulechki. 2009. Sentence Complexity in French: a Corpus-Based Approach. In *Intelligent Information Systems (IIS)*, pages 131–145.

Sowmya Vajjala and Detmar Meurers. 2014. Assessing the Relative Reading Level of Sentence Pairs for Text Simplification. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 288–297. Association for Computational Linguistics.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in Current Text Simplification Research: New Data Can Help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

Xingxing Zhang and Mirella Lapata. 2017. Sentence Simplification with Deep Reinforcement Learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 584–594.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A Monolingual Tree-based Translation Model for Sentence Simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, pages 1353–1361.

A Set of features for simplicity gain

Table 4: Selected features for the definition of the simplicity gain score.

Group	#	ID	Feature	Description
Structural	0	**SL**	Sentence length	*n* of characters comprising a sentence.
	1	**NW**	Number of words	*n* of words comprising a sentence.
	2	**VSL**	Verbal subject length	*n* of words comprising the verbal subject.
	3	**ATL**	Average token length	Average *n* of characters per token in a sentence.
Lexical	4	**CEFR**	CEFR score	Within a sentence, sum of the frequencies of CEFR levels of all non-stop words multiplied by their lexical complexity weight value (Ormaechea and Tsourakis, 2023).
	5	**NE**	Incidence of named entities	*n* of named entities (organizations, people, places, *etc.*) in a sentence.
	6	**LD**	Lexical density	Ratio between the *n* of content words (*i.e.*, nouns, adjectives, adverbs and verbs) and the total *n* of tokens in a sentence.
	7	**TTR**	Type-token ratio	*n* of unique words divided by the total *n* of words in a sentence.
Syntactic	8	**MDT**	Maximum depth tree	Maximum depth of the dependency tree.
	9	**IDT**	Incomplete dependency theory	Average number of incomplete dependencies between the current and next token.
	10	**DLT**	Dependency locality theory	For every head token in a sentence, *n* of discourse referents starting from the current token and ending to its longest leftmost dependent. Values are then combined using an average function.
	11	**LE**	Left embeddedness	*n* of tokens on the left-hand-side of the root verb that are not verbs.
	12	**NND**	Noun nested distance	Average nested distance of all nouns within a phrase that have as ancestor another noun in the dependency tree.
	13	**CC**	Use of coord. clauses	*n* of clauses linked by a coordinating conjunction.
	14	**SC**	Use of subord. clauses	*n* of clauses linked by a subordinating conjunction.
	15	**PR**	Use of parenthetical remarks	*n* of parenthesized information items in a sentence.
	16	**NEG**	Number of negations	*n* of negative adverbs in a sentence (that implies a slower processing with respect to affirmative ones).
	17	**PAS**	Incidence of passive forms	*n* of passive voice verbs in a sentence (that implies a longer reading time with respect to active ones).
	18	**CT**	Incidence of complex tenses	*n* of complex or unusual verb tenses, *i.e.*, those other than infinitive or present, present perfect, imperfect, future indicative.
	19	**IDT-DLT**	Combined IDT-DLT	Sum of IDT-DLT metrics for all tokens in a sentence. Resulting values are then combined using an average function.

B Application of classification and regression models to Wikipedia-Vikidia pairs

Table 5: Applying the triad models to Wikipedia-Vikidia sentence pairs. A gloss in English is provided below each segment for clarity purposes.

	Wikipedia sentence	Vikidia sentence
Pair$_1$	En France, ce lézard est strictement protégé par la loi.	En France, il est protégé par la loi.
Gloss	In France, this lizard is strictly protected by law.	In France, it is protected by law.
AC	`Complex`	`Simple`
RC	`Simplification`	
GR	`0.84`	
Pair$_2$	Praticien précoce et représentant éminent du concept français de la haute gastronomie, il est considéré comme le fondateur de ce style grandiose, recherché à la fois par les cours royales et les nouveaux riches de Paris.	Il est considéré comme l'un des pionniers, sinon le fondateur, de la gastronomie française.
Gloss	As an early practitioner and leading exponent of the French concept of *haute gastronomie*, he is considered the founder of this grandiose style, sought after by both the royal courts and the newly rich of Paris.	He is considered one of the pioneers, if not the founder, of French gastronomy.
AC	`Complex`	`Complex`
RC	`Simplification`	
GR	`2.45`	
Pair$_3$	Makassar ou Macassar est une ville d'Indonésie et la capitale de la province de Sulawesi du Sud.	Macassar ou Makassar est une ville d'Indonésie, située sur l'île de Sulawesi (ou Célèbes), en bordure du détroit du même nom.
Gloss	Makassar or Macassar is a city in Indonesia and the capital of the province of South Sulawesi.	Macassar or Makassar is a city in Indonesia, on the island of Sulawesi (or Celebes), bordering the strait of the same name.
AC	`Simple`	`Complex`
RC	`Complexification`	
GR	`-2.65`	

C Detailed description of the Wikipedia-Vikidia Corpus (WiViCo) dataset

Table 6: Detailed description of WiViCo. We purposely use *texts* and not *sentences* because our dataset includes intersentential examples (*i.e.*, texts comprising more than one sentence).

WiViCo dataset	**Original texts**	**Simpler texts**
# texts	46,525	
# tokens	1,730,277	1,321,139
# types	100,357	73,926
Type/token ratio	5.80	5.60
Average word length	5.27	5.04
Average sentence length	38.63	29.08

Deep Learning-Based Claim Matching with Multiple Negatives Training

Anna Neumann[1], **Dorothea Kolossa**[2], **Robert M. Nickel**[3]
[1]Ruhr-Universität Bochum, Germany
[2]Technische Universität Berlin, Germany
[3]Bucknell University, Lewisburg, PA, USA
anna.neumann1@uni-due.de
dorothea.kolossa@tu-berlin.de
robert.nickel@bucknell.edu

Abstract

Numerous approaches for the implementation of automated fact-checking pipelines have been proposed and reviewed recently (Guo et al., 2022). A key part in these pipelines is a claim matching module that seeks to match new incoming claims with potentially existing, verified claims in a database of completed fact checks. To that end, we propose a modification of the two-stage deep learning-based approach for claim matching which won the CLEF CheckThat! 2022 Subtask 2A Challenge (Shliselberg and Dori-Hacohen, 2022). With our modification, we were able to reduce the error rate of the winning algorithm by more than 20%. This was accomplished by employing a loss function that fuses information from not only a single, but from multiple non-matching (i.e. *negative*) examples into the training process at each iteration.

1 Introduction

Fact-checking became an increasingly important step in journalistic work in response to the proliferation of fake news online. Misinformation on the internet spreads at a speed and scale that makes it more and more difficult for human fact-checkers to react in a timely manner. It is therefore a desirable goal to automate parts of the process. Claim matching is one portion of this process, in which an incoming claim is checked against a database of human-verified claims. The automation of claim matching has received a considerable amount of interest in recent years (Shaar et al., 2022a,b; Kazemi et al., 2021; Nakov et al., 2021).

For the CLEF CheckThat! 2022 Subtask 2A Challenge, Shliselberg and Dori-Hacohen (2022) proposed a two-step pipeline as the winning entry. First, a deep pre-trained language model based on BERT (Devlin et al., 2019) is fine-tuned to generate a selection of candidates of relevant claims from the database. Second, the candidates are re-ranked by fine-tuning a generative language model.

The contribution of this paper is an expansion of the winning method of Shliselberg and Dori-Hacohen (2022) by (1) including additionally mined negative examples into the training objective, (2) investigating a Ranked List Loss (RLL) as an alternative cost function, and (3) expanding the analysis of the proposed scheme by including Mean Average Recall (MAR).

2 Methods

We begin by introducing the employed dataset and the statistical benchmark used for the mining of negative examples. Then, we present the different training objectives for the two stages of the approach. Lastly, we discuss the evaluation metrics.

2.1 Data and Statistical Benchmark

For the dataset, we used the English portion (Subtask 2A) of the dataset provided for the CLEF CheckThat! 2022 Challenge (Nakov et al., 2022) based on verified claims from `Snopes.com`. The dataset consists of 13,835 verified claims, denoted with c for *claim*, and 1,400 input claims, denoted with t for *tweet*. The input claims are divided into 999 tweets for training, 199 tweets for development and 202 tweets for testing. For neural network training, the body of each fact-checked article is tokenized before it is fed into the respective networks.

For our statistical benchmark, we applied a standard BM25[1] ranking algorithm (Robertson and Zaragoza, 2009). The preprocessing for this step includes concatenating the title, subtitle and body of the claim, transforming everything into lowercase, followed by Porter stemming (Porter, 1980). BM25 provides a ranked list of claims for each input tweet. Each non-matching claim is defined as a 'negative' and the matching claim is defined as the 'positive'. The five highest-ranking negatives are mined for our experiments.

[1] *BM* is an abbreviation for *best matching*.

2.2 Candidate Selection

As suggested by Shliselberg and Dori-Hacohen (2022), we use Sentence-T5 (Ni et al., 2022) for the initial candidate selection. It is part of the family of sentence transformers (Reimers and Gurevych, 2019), i.e. deep neural language models based on self-attention mechanisms (Vaswani et al., 2017). It produces sentence embeddings projected on the Euclidean unit circle, which makes the angle between the embeddings a measure of contextual dissimilarity. We use the Multiple Negatives Ranking (*MNR*) Loss (Henderson et al., 2017), which minimizes the distance between the input and positive example and maximizes the distance to all other examples in the batch. Using batches $\mathcal{B} \in \mathcal{D}$ of sets $\mathcal{D} = \{(t_i, c_i^+, c_i^-)\}$ with a tweet t_i, a positive c_i^+ and a negative claim c_i^-, the dot product scoring $S_\theta(t_i, c_i)$ by the specific neural network θ, and a fixed temperature τ, the loss function becomes

$$\mathcal{L}_{MNR}(\mathcal{B}, \theta) = -\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \log \frac{\exp\left(S_\theta(t_i, c_i^+)/\tau\right)}{Q_{\theta,i}}$$

$$\text{with} \quad Q_{\theta,i} = \textstyle\sum_{j \in \mathcal{B}} \exp(S_\theta(t_i, c_j^+)/\tau) + \exp(S_\theta(t_i, c_j^-)/\tau). \qquad (1)$$

As we will see in Section 3, the minimization of the MNR loss can lead to significant performance improvements of the overall system when it is expanded by mining not one but multiple negatives for every paired example. This gives the model potentially even more context, as the tweet is compared to every claim in the batch. We used up to five negatives that ranked highest in the BM25 run. For sets $\mathcal{D} = \{(t_i, c_i^+, c_{i,1}^-, c_{i,2}^-, c_{i,3}^-, c_{i,4}^-, c_{i,5}^-)\}$ the expanded loss is defined by

$$\mathcal{L}_{MNR}(\mathcal{B}, \theta) = -\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \log \frac{\exp(S_\theta(t_i, c_i^+)/\tau)}{\sum_{j \in \mathcal{B}} \mathcal{M}_{\theta,i,j}}$$

$$\text{with} \quad \mathcal{M}_{\theta,i,j} = \exp(S_\theta(t_i, c_j^+)/\tau) + \textstyle\sum_{k=1}^{5} \exp(S_\theta(t_i, c_{j,k}^-)/\tau). \qquad (2)$$

2.3 Generative Re-Ranking

For the second step, we, again, follow closely the setup proposed by Shliselberg and Dori-Hacohen (2022). The fine-tuned S-T5 network from the first step is used to generate ranked lists of the five most similar claims to each input tweet. The combination of the claim and each input tweet is then employed separately to fine-tune the generative deep language model GPTNeo[2] (Black et al., 2022). The generative model has the capacity to calculate the conditional probability $p(t|c)$ for tweet and claim pairs, which, in turn, can be utilized to re-rank the given list of five tweet/claim pairs provided by the preceding stage. The tweets and claims therefore have to be converted into prompts with beginning-of-sentence $< bos >$ and end-of-sentence $< eos >$ tokens: $< bos > c < eos >< bos > t < eos >$.

For the fine-tuning of GPTNeo, we considered a number of loss functions, primarily motivated by the loss types recommended in the commensurate literature. We expanded on the list of losses considered by Shliselberg and Dori-Hacohen (2022) by also including the standard ranked list loss (*RLL*) into our analysis. The RLL (Nogueira dos Santos et al., 2020) is defined by

$$\mathcal{L}_{RLL}(\mathcal{B}, \theta) = \sum_{i \in \mathcal{B}} \max\{0, \lambda - \log p_\theta(t_i|c_i^+) + \log p_\theta(t_i|c_i^-)\}, \qquad (3)$$

in which the notation p_θ denotes the dependence of the probability estimate on the network parameters θ. The hinge margin λ is a hyperparameter of the training procedure. The also employed NL3U loss (Nogueira dos Santos et al., 2020) is based on the negative log-likelihood loss (Lesota et al., 2021). It incorporates the *unlikelihood probability*[3] of the negative claim:

$$\mathcal{L}_{NL3U}(\mathcal{B}, \theta) = -\sum_{i \in \mathcal{B}} \log p_\theta(t_i|c_i^+) + \log(1 - p_\theta(t_i|c_i^-)). \qquad (4)$$

Shliselberg and Dori-Hacohen (2022) also introduced the mixed objective

$$\mathcal{L}_{Mix} = \mathcal{L}_{MI_1} + \mathcal{L}_{MI_2} + \mathcal{L}_{NL3U}, \qquad (5)$$

which utilizes a hinged prior mutual information loss ($\mathcal{L}_{MI_1}$) and a posterior-based hinged mutual information loss ($\mathcal{L}_{MI_2}$). The loss $\mathcal{L}_{MI_1}$ maximizes mutual information but reverts back to the maximum likelihood estimate above a threshold λ:

$$\mathcal{L}_{MI_1} = \begin{cases} \mathcal{K}_\theta^{\text{MI}} & \text{if } -\log \frac{p_\theta(t|c)}{p_\theta(t)} < \lambda \\ \mathcal{K}_\theta^{\text{MLE}} & \text{otherwise} \end{cases} \qquad (6)$$

with

$$\begin{aligned} \mathcal{K}_\theta^{\text{MI}} &= \text{E}_{T,C}[-\log p_\theta(t|c) + \log p_\theta(t)] \\ \mathcal{K}_\theta^{\text{MLE}} &= \text{E}_{T,C}[-\log p_\theta(t|c)] \end{aligned}$$

[2] GPTNeo contains on the order of 1.3 billion parameters.

[3] The *unlikelihood probability* is defined as one minus the probability.

To also model the posterior, the input order is flipped and $\mathcal{L}_{MI_2}$ is defined as:

$$\mathcal{L}_{MI_2} = \mathrm{E}_{C,T}[max(0, \lambda - \log p_\theta(c|t) + \log p_\theta(c))] \tag{7}$$

We utilize $\mathcal{L}_{RLL}$, $\mathcal{L}_{NL3U}$, $\mathcal{L}_{Mix}$ and a sum of both mutual information losses, i.e.

$$\mathcal{L}_{MutInf} = \mathcal{L}_{MI_1} + \mathcal{L}_{MI_2}, \tag{8}$$

as training objectives to fine-tune the generative model.

2.4 Evaluation Metrics

The CheckThat! Challenge uses Mean Average Precision (*MAP*) for evaluation. In addition to MAP scores, we also considered the Mean Average Recall (*MAR*) in our analysis. Average Recall at length k is defined as

$$\text{AR@}k\{R, g\} = \sum_{i=1}^{k} \mathbb{1}[R[i] == g] \tag{9}$$

for a ranked list R with one gold label g, using $\mathbb{1}$ as an indicator function. MAR is the mean over all ranked lists and label pairs denoted as Ω:

$$\text{MAR@}k\{\Omega\} = \frac{1}{|\Omega|} \sum_{R,g \in \Omega} \text{AR@}k\{R, g\} \tag{10}$$

We are using the standard definition of MAP@k as the mean value over the average precision AP@$k\{R, g\} = \sum_{i=1}^{k} \mathbb{1}[R[i] == g]\frac{1}{i}$. MAP and MAR values are bounded between zero and one. MAP@1 equals MAR@1 by definition. We therefore only report MAP@1 scores.

3 Experiments

In our experiments we generated five S-T5 models through fine-tuning: One with an MNR loss with one negative, one with an MNR loss with two negatives, and so forth, up to one with an MNR loss with five negatives. Each S-T5 model is then paired with four GPTNeo models, one for each of the four loss functions described in Section 2.3, to create a total number of 20 systems. The ranked lists generated by the S-T5 models are used to fine-tune the respectively paired GPTNeo models. All training was performed on a server with two NVIDIA RTX A6000 GPUs with 48 GB memory each. We report MAP@1, MAP@5, and MAR@5 scores in all our cases. All evaluations are performed on the test set defined by the CheckThat! Challenge. Finally, we compare top performing methods to the BM25 benchmark.

3.1 Candidate Selection

S-T5 is fine-tuned using a batch size of 3, the AdamW optimizer with a constant learning rate of 5e-6, an MNR loss temperature τ of 0.1 and a maximum of 128 tokens for each input for a single epoch. Results are presented in Table 1. The respective highest values in each column are highlighted in bold face.

#Neg	MAP@1	MAP@5	MAR@5
1	0.896	0.932	0.975
2	0.896	0.933	**0.980**
3	**0.901**	**0.936**	**0.980**
4	0.896	0.934	**0.980**
5	0.891	0.931	**0.980**

Table 1: Evaluation of the S-T5 MNR Fine-Tuning.

Both, MAP@1 and MAP@5, peak for a training with three negatives and then both decline again for a training with four and five negatives. The improvement that a training with three negatives affords over a training with one negative is on the order of 0.5 percentage points in both cases. This finding supports our hypothesis that training with more negatives provides better context for the model. Yet, training with too many negatives appears to put too much weight on the rejection of negatives and too little weight on the support of positives. The MAR@5 values increase slightly for two negatives and then stay constant at 0.980.

3.2 Generative Re-Ranking

Fine-tuning the re-rankers for one epoch includes a batch size of 1, a maximum of 256 input tokens for padded prompts, a hinge margin $\lambda = 2$ and the AdamW optimizer with a learning rate of 2e-5. Our system with one-negative-training is essentially the system proposed by Shliselberg and Dori-Hacohen (2022). The MAP@5 score we obtained for the one-negative/mixed case matches the result reported by them closely. We attribute slight deviations to differences in random initialization, differing batch sizes due to computational limits and batch shuffling. We omitted the MAR@5 results, since these did not change and stayed constant at 0.980 for every number of negatives above one. The MAR@5 results for one negative all came out to 0.975. The best-performing loss for each base model in each column is highlighted in bold. It is apparent that the mixed approach performed best for all candidate selection models and that the RLL approach

#Neg	Loss	MAP@1	MAP@5
1	Mixed	**0.921**	**0.947**
	NL3U	0.911	0.941
	MutInf	0.896	0.935
	RLL	0.658	0.778
2	Mixed	**0.936**	**0.958**
	NL3U	0.901	0.938
	MutInf	0.891	0.933
	RLL	0.757	0.850
3	Mixed	**0.926**	**0.953**
	NL3U	0.921	0.949
	MutInf	0.891	0.933
	RLL	0.223	0.475
4	Mixed	**0.926**	**0.953**
	NL3U	0.921	0.947
	MutInf	0.886	0.931
	RLL	0.871	0.925
5	Mixed	**0.926**	**0.953**
	NL3U	0.916	0.945
	MutInf	0.906	0.942
	RLL	0.183	0.421

Table 2: Evaluations over the test set of the CLEF CheckThat! 2022 Subtask 2A Challenge with GPTNeo re-ranking for four training objectives.

performed the worst. The NL3U loss alone performs significantly better than RLL and achieves a peak performance in MAP@1 and MAP@5 for three negatives. The mutual information loss consistently performs a bit worse than NL3U alone and achieves its best performance for five negatives. The mixed loss consistently outperforms other losses and peaks for two negatives with a MAP@1 value of 0.936 and a MAP@5 value of 0.958. When compared to the MAP@5 value of 0.947 for the training with a single negative, it can be seen that the error rate, when defined as one minus MAP@5, is reduced by over 20% with the proposed multiple negatives training. We attribute the superior performance of the mixed loss training to the fact that it incorporates different aspects of text similarity, measuring mutual information on the one side and contrasting it with information about negative examples on the other side.

3.3 Discussion

We present the BM25 evaluation, the best-performing candidate selection model and the best-performing generative re-ranking model in Table 3. The BM25 method already provided comparatively high scores with a MAP@5 value of 0.852 and a MAR@5 value of 0.936. The best-performing S-T5 network, fine-tuned with a three-negatives MNR loss, outperforms the BM25 baseline by more than eight percentage points with a MAP@5 of 0.936. The MAP@1 value increases by over ten points and the MAR@5 value by over four points. The fine-tuned GPTNeo system based on a two-negatives S-T5 model with a mixed objective yields the best performance with a MAP@5 value of 0.958. It outperforms the reference model with mixed loss and one-negative-training by about one percentage point and the best candidate selection model by over two percentage points.

Model	MAP@1	MAP@5	MAR@5
BM25	0.797	0.852	0.936
S-T5$_{3N}$	0.901	0.936	**0.980**
GPT$_{Mix,2N}$	**0.936**	**0.958**	**0.980**

Table 3: Performance summary for experiments over the test set from the CLEF CheckThat! 2022 Subtask 2A Challenge.

4 Conclusions and Future Work

The winning algorithm of the CLEF CheckThat! 2022 Challenge for claim matching (Subtask 2A) consists of a two step process: (1) candidate selection and (2) generative re-ranking. Both steps are achieved with fine-tuned deep neutral networks. We generalized the loss function used in the fine-tuning of the candidate selection network by including not just one negative example but multiple negative examples. Through experimentation with various configurations and loss functions we were able to create an overall system that improves the MAP@1 and MAP@5 scores by over one percentage points each, leading to an effective reduction in error rate of around 20%. Future work may include an incorporation of other, more powerful large language models (*LLMs*) in lieu of GPTNeo.

Acknowledgements

The work on this paper was performed as part of a master's thesis at the Ruhr-University Bochum in support of the project "noFake", funded by the German Federal Ministry of Research and Education (BMBF) under Award Identifier F16KIS1519. Our implementations are based on the code provided by Shliselberg and Dori-Hacohen (2022) at https://github.com/RIET-lab/GenerativeClaimMatchingPipeline.

References

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-NeoX-20B: An Open-Source Autoregressive Language Model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, Dublin, Ireland. 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.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Matthew L. Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient Natural Language Response Suggestion for Smart Reply. Computing Research Repository, arXiv: 1705.00652.

Ashkan Kazemi, Kiran Garimella, Devin Gaffney, and Scott Hale. 2021. Claim Matching Beyond English to Scale Global Fact-Checking. 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 4504–4517, Online. Association for Computational Linguistics.

Oleg Lesota, Navid Rekabsaz, Daniel Cohen, Klaus Antonius Grasserbauer, Carsten Eickhoff, and Markus Schedl. 2021. A modern perspective on query likelihood with deep generative retrieval models. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 185–195, New York, USA. Association for Computing Machinery.

Preslav Nakov, Alberto Barrón-Cedeño, Giovanni da San Martino, Firoj Alam, Julia Maria Struß, Thomas Mandl, Rubén Míguez, Tommaso Caselli, Mucahid Kutlu, Wajdi Zaghouani, Chengkai Li, Shaden Shaar, Gautam Kishore Shahi, Hamdy Mubarak, Alex Nikolov, Nikolay Babulkov, Yavuz Selim Kartal, Michael Wiegand, Melanie Siegel, and Juliane Köhler. 2022. Overview of the clef–2022 checkthat! lab on fighting the covid-19 infodemic and fake news detection. In *Proceedings of the 13th International Conference of the CLEF Association*, pages 495–520, Bologna, Italy.

Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021. Automated Fact-Checking for Assisting Human Fact-Checkers. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 4551–4558, Montreal, Canada. International Joint Conferences on Artificial Intelligence Organization.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-T5: Scalable Sentence Encoders from Pretrained Text-to-Text Models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

Cicero Nogueira dos Santos, Xiaofei Ma, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Beyond [CLS] through Ranking by Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1722–1727, Online. Association for Computational Linguistics.

Martin F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

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

Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Shaden Shaar, Firoj Alam, Giovanni Da San Martino, and Preslav Nakov. 2022a. The Role of Context in Detecting Previously Fact-Checked Claims. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1619–1631, Seattle, USA. Association for Computational Linguistics.

Shaden Shaar, Nikola Georgiev, Firoj Alam, Giovanni Da San Martino, Aisha Mohamed, and Preslav Nakov. 2022b. Assisting the Human Fact-Checkers: Detecting All Previously Fact-Checked Claims in a Document. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2069–2080, Abu Dhabi, UAE. Association for Computational Linguistics.

Michael Shliselberg and Shiri Dori-Hacohen. 2022. RIET Lab at CheckThat! 2022: Improving decoder based re-ranking for claim matching. *Proceedings of the Working Notes of CLEF 2022 - Conference and Labs of the Evaluation Forum*, 3180:671–678.

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

Exploring BERT Models for Part-of-Speech Tagging in the Algerian Dialect: A Comprehensive Study

Mohamed Amine Cheragui[1], **Abdelhalim Hafedh Dahou**[2] and **Amin Abdedaiem**[1]
[1] Mathematics and Computer Science Department Ahmed Draia University Adrar - Algeria
[2] GESIS-Leibniz-Institute for the Social Science Cologne - Germany
m_cheragui@univ-adrar.edu.dz
abdelhalim.dahou@gesis.org
aminabdedaiem@gmail.com

Abstract

Social media have given a new impetus to natural language processing, especially for Arabic, by orienting research towards varieties of languages called dialects, which are less prestigious linguistically than Modern Standard Arabic (MSA) but are becoming more and more important as informal communication channels through different platforms: emails, blogs, discussion forums and SMS, offering a fertile research area. Part-of-speech (POS) tagging holds significant importance in various natural language processing applications, particularly in languages with complex morphological characteristics like Arabic. While a substantial part of research has concentrated on POS tagging for MSA, studies on dialects are scarce due to limited linguistic resources. This paper aims to showcase our efforts in advancing a morphosyntactic tagger tailored for the Algerian dialect. We accomplish this through a series of experiments employing a pre-trained Arabic transformer model, fine-tuned on various writing styles of the Algerian dialect commonly encountered in social media and everyday communication. Our proposed model outperforms previous state-of-the-art models, achieving an accuracy rate of 87% for Dz writing style and 83% for Arabizi writing style.

1 Introduction

Recognizing the nature of a word in a context (classification of words according to their behaviour in language) is a non-trivial task in natural language processing (NLP). Indeed, making a machine capable of knowing the linguistic category of a word requires the implementation of sophisticated methods, in particular for ambiguous words, i.e. those that may belong to several morphosyntactic categories. Such automatic tools are called Part of Speech tagger.

POS tagging is a fundamental task for NLP, on which complex processes such as information extraction or machine translation, syntactic analysis, etc., are often based. By definition, POS tagging is a process that assigns a morpho-syntactic tag to each word in a text specifying, in particular, grammatical category, gender, number, tense, and mode (Nerabie et al., 2021).

Arabic language represents a real challenge in terms of POS tagging, mainly due to its particular morphological system, both rich and complex as a consequence of two linguistic phenomena which are inflection and derivation, which make the process of recognizing the parts of speech a tedious task (Habash and Rambow, 2005). Arabic is a language that is spoken by a population of about 428 million people [1] and extends over a huge geographical area from the Arabian Gulf to the Atlantic, spread over 22 countries. This geographical expansion has contributed to the emergence of several variants of the Arabic language called "aammiyya" dialect (colloquial Arabic) as opposed to fusha (literary Arabic). Although these dialects share some common characteristics, they differ on many linguistic levels from standard Arabic (Katz and Diab, 2011).

According to (Habash, 2010), we can enumerate 30 variants of Arabic dialects. The interest in this variant of the language, in despite of the difficulties it presents, in particular the lack of orthographic normalization and standardization, is due to its expansion in terms of use, especially in social media, offering a research field with many challenges.

In this paper, we outline our approach for the development of morphosyntactic POS tagging in the context of one of the most prevalent dialects found on social networks—the Algerian dialect. We achieve this by:

- Assessing and exploring several models based on the BERT architecture (AraBERT v0.2-base, AraBERT v0.2-Twitter-base, Dziribert,

[1] World Population Review. Arab Countries 2020. Washington, DC. https://worldpopulationreview.com/countries/arab-countries/. Accessed August 9, 2023

MARBERT and m-BERT).

- Tackle different Algerian writing styles including: Arabic letters (Dz), Latin characters (Arabizi), and code-switching.

- Addressing the research question of the performance achieved by models trained solely on MSA when tested in various writing styles of the Algerian dialect.

The paper is organized into six sections. Section 1 introduces the research problem, while Section 2 provides a comprehensive review of related works in the field of Arabic dialect POS tagging. Our contribution is detailed in Section 3, and Section 4 offers insights into the dataset employed across various stages of experimentation. The experimental results are deliberated upon in Section 5, and, in conclusion, Section 6 summarizes our findings and outlines a vision for future research endeavors.

2 Related work

The Part of Speech tagging (POS) is a process that consists in assigning to each recognized entity a set of morphosyntactic features (Albared et al., 2011). This process has a crucial impact on the performance of several tools (Chunkers and Parsers, etc) and applications (Machine translation, Information retrieval, Text summarization, Sentiment analysis, etc) in NLP. For the MSA, POS tagging has been the subject of several works involving different approaches: rule-based, stochastic, and machine learning. However, for the Arabic dialect, research is scarce, due mainly to two factors: the lack of resources (corpus and tools: morphological analyzers, tokenizers, etc.), and there is no orthographic standards. The Dialectical Arabic (DA) POS tagging techniques follow two principal approaches. The first approach suggests using MSA resources and a few DA resources to create a POS tagger (Salloum and Habash, 2011) and the second intends to start from scratch.

Boujelbane et al. (Boujelbane et al., 2014), Retrained an MSA tagger which is the Stanford POS Tagger (Toutanvoa and Manning, 2000), using a corpus derived from a translation of the MSA Treebank into Tunisian Dialect, and adapt it to perform the tagging on the Tunisian dialect. The POS tagger set up achieved an accuracy of 78,5%.

Al-Sabbagh and Girju (Al-Sabbagh and Girju, 2012a), described a POS tagging based on the Brill's Transformation-Based Learning (Brill, 1994), for the Egyptian Dialect. For training and testing, the authors have built a golden corpus that contains 22,834 tweets, 423,691 tokens and 70,163 types. The tool obtained an F-measure score of 87.6%.

Baniata et al. (Baniata et al., 2018), presented a Bidirectional Long Short-Term Memory (Bi-LSTM)—Conditional Random Fields (CRF) segment-level Arabic Dialect POS tagger model for the Levantine Arabic (spoken variety of widely used in Jordan, Syria, Palestine and Lebanon) and Maghrebi (Morocco, Algeria and Tunisia), which will be integrated into the Multitask Neural Machine Translation (NMT) model. For the experimental part, they used the dataset described in (Darwish et al., 2018), which contains 350 tweets for four major Arabic dialects. Their POS tagger achieved an accuracy of 98% and 99% for the Levantine and Maghrebi dialect respectively.

Darwish et al. (Darwish et al., 2018), proposed a POS tagger for several Dialects (Egyptian, Levantine, Gulf, and Maghrebi), based on CRF. The authors have defined 03 features including clitic n-grams, clitic metatypes, and stem templates. For training and testing, a dataset covering all 04 dialects was built from 350 tweets for each dialect. For the results, the authors proposed 03 learning setups: the first one consists on treating each dialect alone, the model obtained the following results: 92.9% for Egyptian, 87.9% for Levantine, 87.8% for Gulf, and 88.3% for Maghrebi. In the second one, the dialects joined, the model gave the following results: 93.2% for Egyptian, 88.6% for Levantine, 87.2% for Gulf, and 87.7% for Maghrebi. In the third configuration, the dialects combined with the MSA, the model gave the following results: 93.4% for the Egyptian, 88.6% for the Levantine, 87.4% for the Gulf, and 87.6% for the Maghrebi.

Alharbi et al. (Alharbi et al., 2018), designed a Gulf Arabic (GA) POS taggers using two approaches: Support Vector Machine (SVM) classifier and BI-LSTM. For the SVM classifier, they defined 03 set features: Clitic features, Probabilistic features and Binary features. For the second Bi-LSTM classifier, the authors used Java Neural Network (JNN) toolkit for language modelling and POS tagging (Ling et al., 2015). The input of the network is a sequence of features: clitic, meta type, and/or stem template. For the experimental part, they used a gold annotated dataset which is built using gold segmented GA tweets taken from (Samih et al., 2017). Dataset consists of 343 Tweets with 6,844 tokens and 10,255 clitics. For the tag sets, they adopted the same one proposed by (Darwish et al., 2017) which composed of 18 tag sets. In addition, they added 04 others new tags for twitter specific data including: MENTION, URL, HASH, and EMOT. The two models SVM and Bi-LSTM obtained respectively an accuracy score of 85.96% and 91.2%.

Duh and Kirchhoff (Duh and Kirchhoff, 2005), built a Levantine and Egyptian POS tagger. They used the Buckwalter Morphological Analyzer designed for MSA, the LDC MSA Treebank corpus and some dialectal resources (the CallHome Egyptian Colloquial Arabic corpus ECA, the LDC Levantine Arabic corpus) in combination with unsupervised learning algorithms. The author's contribution consisted of bootstrap the Hidden Markov Models (HMM) tagger using POS information from the morphological analyzer. The developed tool obtained an accuracy of 70.88%.

Darwish et al. (Darwish et al., 2020), built a multi-dialectal POS tagger (covering Egyptian, Levantine, Gulf, and Maghrebi dialects) based on two approaches: CRF classifier combined with linguistic features (stem templates and clitic metatypes), word clusters from a large unlabeled tweet corpus, and automatic dialect identification; while the second combines word-based and character-based representations in a deep neural network with stacked layers of convolutional and recurrent networks with a CRF output layer. They achieve a combined accuracy of 92.4% across all dialects, with per dialect results ranging between 90.2% and 95.4%.

Hamdi et al. (Hamdi et al., 2015), developed a POS tagger for the Tunisian dialect. Their idea was to convert Tunisian into an approximate form of MSA, called pseudo MSA, and use an existing MSA POS tagger. The output produced is then projected back on the Tunisian text. The system operates through a three steps process: firstly, conversion is performed using MAGEAD, a morphological analyzer/generator; secondly, disambiguation is carried out; and finally, POS tagging is accomplished using HMM. For the evaluation, they used a transcribed and annotated corpus of 805 sentences containing 10,746 tokens and 2,455 types. The system achieved an accuracy of 89%.

AlKhwiter and Al-Twairesh (AlKhwiter and Al-Twairesh, 2021), proposed two supervised POS taggers for both MSA and the Gulf Dialect that are developed based on two approaches including CRF and Bi-LSTM. For the experimentation, the authors built three annotated datasets named Mixed, MSA, and GLF containing respectively 3, 1000, and 1000 Arabic tweets. As a result for the Gulf Dialect, the CRF and Bi-LSTM achieved an accuracy of 90% and 95% respectively.

Inoue et al. (Inoue et al., 2022), proposed morphosyntactic tagging model for three Arabic dialects: Gulf, Egyptian and Levantine, based on Pre-trained Language Model (CAMeLBERT-Mix) with two variants Factored and Unfactored Tags. The authors report that they obtained an accuracy of 94.6% for the Egyptian, 97.9% for Gulf, and 94.0% for Levantine, using respectively, ARZTB, Gumar Corpus, and Curras Corpus.

Pasha et al. (Pasha et al., 2014), presented MADAMIRA, which is a combined version of previously developed tools: MADA (Habash et al., 2009) and AMIRA (Diab, 2009), based on SVM. It provides various functions, such as tokenization, POS tagging and phrase chunking. The tool was trained on the Penn Arabic Treebank corpus for MSA and the Egyptian Arabic Treebanks for the Egyptian dialect. The performance of MADAMIRA was evaluated through a blind test dataset, and achieved an accuracy rate of 92.4% for the Egyptian Dialect.

3 Contribution

As previously stated, POS tagging is a preprocessing phase and an essential block in numerous NLP applications that require the syntactic category for each text token. The related work section demonstrates that the Arabic language has less work than the other language owing to its highly inflectional structure. Furthermore, the majority of Arabic works in POS and cutting-edge POS taggers are dedicated to the MSA variant, which is the formal language used in journalism and government administrations. DA is the more casual Arabic version used in everyday life, got less attention by researchers due to its great complexity when compared to the MSA.

With the passage of time, DA grew more frequently utilized, particularly in social media, and MSA POS taggers struggled to acquire good results when applying for DA texts (Pasha et al., 2014). Our contribution focused on developing a dialec-

Table 1: Arabic Dialect Annotated (POS) Corpus.

Corpus	Dialect	Token	Annotation
YADAC (Al-Sabbagh and Girju, 2012b)	EGY	6 M	FST and Manually
ARZATB (Fashwan and Alansary, 2022)	EGY	475 K	CALIMA and Manually
NArabizi (Seddah et al., 2020)	ALG	19770	Manually
LATB (Maamouri et al., 2006)	LEV	26 K	/
Gumar (Khalifa et al., 2016)	GULF	112 M	MADAMIRA
Curras (Jarrar et al., 2014)	PAL	43 K	DIWAN and Manually
Baladi (Al-Haff et al., 2022)	LEB	9.6 K	Manually (AnnoSheet)
MOR (Al-Shargi et al., 2016)	MOR	64170	DIWAN
YEMS (Al-Shargi et al., 2016)	YEM	32445	DIWAN

tal POS tagger for one of the more broadly used Arabic dialects in social media, the Algerian dialect, using an Arabic pretrained model based on the BERT architecture and fine-tuned on different writing styles of the Algerian dialect found in social media and used in everyday life. Furthermore, this study will investigate whether a POS tagger trained on the Algerian dialect may outperform an MSA POS tagger.

3.1 Transformers and BERT Models

With the rise of the RNN model and its variations problems, recently developed techniques were proposed to overcome those limitations including the Transformer-based architecture built based on the attention mechanism (Vaswani et al., 2017). The Transformer-based model is known for their high performance in terms of learning contextualized text representation. BERT (stands for Bidirectional Encoder Representations from Transformers) is one of the most popular NLP models that utilizes a transformer at its core and which achieved state of the art performance on many NLP tasks including Classification, Question Answering, and NER Tagging when it was first introduced. Contextualized text or word representation means that the embeddings of a word is not static. That is, they depend on the context of words around it. So in a sentence like 'خويا روح نيشان، دوك تلقاه' 'my brother go forward, you will find it', and the other sentence 'عندك صح راك نيشان' 'you are right!', the two embeddings of the word 'نيشان' will be different, which in the first means "forward" and in the second sentence means 'right'. While directional models in the past like LSTM read the text input sequentially (left-to-right or right-to-left), the Transformer actually reads the entire sequence of words at once and thus is considered bidirectional.

The developed models for English such as BERT, DistilBERT (Sanh et al., 2019), BART (Lewis et al., 2020), were adopted and used in Arabic NLP showing remarkable performance. In this study, we will evaluate the performance of those Arabic pretrained models and take the one that achieves high performance. For instance, the AraBERT (Antoun et al., 2020) a pre-trained model on large MSA and dialects data from Wikipedia and Twitter, MARBERT trained on Maghrebi dialectes data representing coutries such as Algeria, Morocco, and Tunisia (Abdul-Mageed et al., 2021), DziriBERT trained on Algerian dialect (Abdaoui et al., 2022), mBERT (Pires et al., 2019) trained on the top 104 languages including Arabic and its dialects with the largest Wikipedia data. Table 2 presents a comparison between those Arabic pre-trained models in terms of size, dataset and vocab.

3.2 Fine-tuning Models

As mentioned previously, BERT is a big neural network architecture, with a huge number of parameters, that can range from 100 million to over 300 million. Given this complexity, training a BERT model from scratch, particularly on a small dataset, predisposes it to overfitting due to the disproportionate ratio of parameters to data points. Consequently, it is more effective to utilize a pre-trained BERT model, which has been subjected to rigorous training on a voluminous dataset, as an initial framework. Then further train the model on our relatively smaller dataset and this process is known as model fine-tuning. This mechanism can be done in three ways, the first is to train the entire architecture of the pre-trained model, the second consists of training some layers while freezing others, and the third one freezes the entire architecture and trains

Table 2: Used Arabic pre-trained models.

Model	Size (params)	DataSet (nwords)	Vocab size
AraBERT v0.2-base	136M	8.6 Billion	64K
AraBERT v0.2-Twitter-base	136M	8.6 Billion + 60 Million Multi-Dialect Tweets	/
Dziribert	124M	1 million tweets	50k
MARBERT	163M	6.2 Billion	100k
mBERT	110M	1.5 Billion	106k

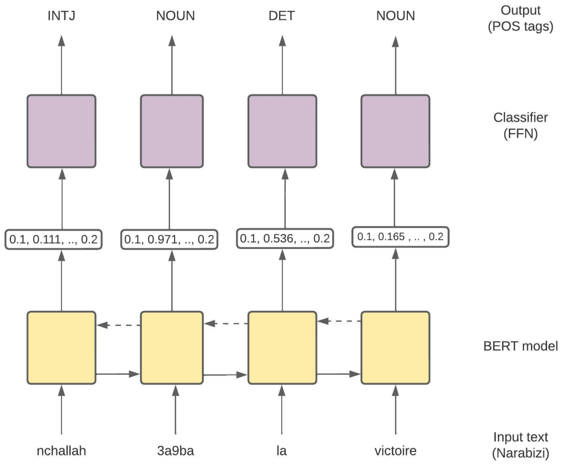


Figure 1: Model fine-tuning architecture.

just the classification layer. In our study, we train the entire pre-trained model on our dataset and feed the output to a softmax layer. In this case, the error is back-propagated through the entire architecture and the pre-trained weights of the model are updated based on the new dataset. Figure 1 describes the overall architecture of fine-tuning a bert model on POS tasks.

As shown in figure 1, our model is composed of an Arabic pre-trained BERT model and a simple linear layer. We can think of the BERT as an embedding layer and all we do is add a linear layer on top of these embeddings to predict the tag for each token in the input sequence. The yellow squares were the embeddings provided by the pretrained BERT model. All inputs are passed to BERT at the same time. The arrows between the BERT embeddings indicate how BERT does not calculate embeddings for each tokens individually, but the embeddings are actually based on the other tokens within the sequence which give us at the end a contextualized embedding. Finally, we fed the output of the pretrained BERT to the Linear layer of size (embedding_dim x n_outputs) and added a softmax layer on top to predict the POS Tagging like predicting noun, verb, or adjective.

4 Dataset

The dataset employed in this study, as delineated in (Touileb and Barnes, 2021), originates from the NArabizi treebank detailed by (Seddah et al., 2020). It encompasses a corpus of 1,300 Arabizi sentences sourced from an Algerian newspaper's web forum and an additional 200 sentences derived from song lyrics manually collated from various online platforms. Each sentence within this dataset is annotated across five distinct layers: tokenization, morphological analysis, code-switching identification, syntactic structure, and translation into French.

This dataset was further augmented to include two additional annotations for each token in the Arabizi sentences. The first enhancement involves the transliteration of each Arabizi token into the Arabic script, with the resultant dataset designated as 'DZ'. The second augmentation entails the transliteration of each Arabizi token into a code-switched script—either Arabic or Latin—depending on the token's origin, thus forming the code-switched dataset. As (Touileb and Barnes, 2021) assert, these annotations were meticulously conducted by bilingual native speakers of Algerian Arabic and French, adhering to standardized guidelines. Table 4, extracted from dataset' paper, exemplifies these stylistic variations within the dataset.

In the preceding sections, we outlined the focus of this study, which centers on evaluating the performance efficacy of a POS tagger specifically trained on the Algerian dialect. This investigation aims to ascertain whether such a dialect-specific POS tagger can surpass the performance of a MSA POS tagger. To facilitate a comprehensive and objective comparison, an MSA dataset, specifically curated for POS tagging, will be employed. For that, the

MSA dataset used is available in UD [2] (Zeman et al., 2020) and also has the same labels as the NArabizi dataset just with difference in terms of number of sentences and the average length. For the NArabizi, we have *19,770* tokens, *1,276* sentences with an average of *16.1* tokens. In MSA, we found *262,803* tokens, *8,664* sentences with an average of *42.3* tokens. Table 3 describe the distribution of POS tags in both datasets.

Table 3: Distribution of POS tags in both datasets in terms of numbers.

Category	**NArabizi**	**UD (MSA)**
NOUN	1981	10588
VERB	1819	3805
ADJ	624	4968
PROPN	552	1052
PRON	263	64
ADV	260	134
ADP	157	156
INTJ	120	5
DET	87	76
SCONJ	56	6
PART	36	36
PUNCT	36	18
CCONJ	32	95
NUM	9	994

5 Experiments and Evaluation

This section presents the experimental setup used, the experiments carried out as part of our research with a comparison between our best model and previous works.

5.1 Experimental setup

For the performance measures, the models are evaluated by calculating the precision, recall, Accuracy and F1-score of their output on the test dataset. Precision and recall are often used metrics to provide more accurate outcomes as well as to provide more information to the expert about the model's behavior, particularly in multi-class classification. To accelerate the training and testing phase, all of them were carried out using the Google Colab platform with a GPU Tesla P100-PCIE-16GB and the Hugging Face Transformers library (Wolf et al., 2020), was used in all our experiments. Using the test dataset, we fine-tuned the hyper-parameter to find the optimal configuration for each pre-trained model in order to achieve the best results. The hyper-parameter settings for each model are listed in table 5.

5.2 Results and discussion

This study comprises two experimental series, the first series of experiments looks at the performance of each Arabic pre-trained model indicated above on the Algerian dataset. The second trial series investigates how much performance can be obtained by applying MSA-specific models to adapt to Algerian dialect.

5.2.1 Experimental series 1

We ran three separate trials in this experimental series to assess the performance of each model on the dataset. The experiments are as follows: first, focus on the Dz writing style, which only uses Arabic letters; second, on the Arabizi style, which utilizes Latin characters; and third, on the code-switched style, which combines Latin and Arabic characters. Table 6 provides the results for each model on the three writing styles in terms of accuracy and F1 score. Finally, we compare our best-obtained results to previously published research in table 7.

The findings shown in table 6 demonstrated that the models performed well on the three writing styles of the Algerian dialect. The AraBERT twitter model obtained the highest results for the first style of writing that employs only Arabic words (Dz), with an F1 score of 84.6%, followed by the AraBERT base and DziriBERT models. This accomplishment is due to the variety of text sources and the volume of MSA and dialectal data encountered in the pre-training phase, which allows the model to represent the majority of Arabic words while avoiding out-of-vocabulary words. In the Arabizi writing style, the DziriBERT model exhibited superior performance, attaining the highest F1-score of 79.5%. Following closely behind was the mBERT model, which was trained across multiple languages, including French, predominantly used by the Algerian community especially in the Arabizi writing style. This multilingual training contributed to its commendable results. Even though DziriBERT's vocabulary is limited and it has seen less text in the pre-training phase compared to the other models, this demonstrates that pre-training a model for one dialect on a small training set

[2] Universal dependencies Corpus: https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-3226.

Table 4: Examples of the writing styles exist in the dataset.

Arabizi	ycombati la misere li las9at fina welat kiste
Arabic transliteration (Dz)	يكومباطي لا ميزار لي لسقت فينا ولات كيست
Code-switched transliteration	يكومباطي la misère لي لسقت فينا ولات kyste
English translation	He fights the misery that sticks to us and which has become a cyst

Table 5: Hyper-parameters values for each used model.

Models	**Epochs**	**learning rate**	**warmup steps**	**seed**
AraBERT v0.2-base	20	5e-5	42	42
AraBERT v0.2-Twitter-base	20	5e-5	42	42
Dziribert	15	5e-5	0	42
MARBERT	15	3e-5	42	42
mBERT	15	5e-5	42	666

Table 6: Performance results on Algerian dataset for the three writing styles.

	Dz		**Arabizi**		**Code-switching**	
Model	**Accuracy**	**F1 score**	**Accuracy**	**F1 score**	**Accuracy**	**F1 score**
AraBERT base v0.2	0.871	0.845	0.801	0.764	0.893	0.873
AraBERT v02-twitter	0.867	**0.846**	0.795	0.752	0.892	0.869
MARBERT	0.861	0.834	0.795	0.757	0.892	0.864
DziriBERT	0.866	0.840	0.831	**0.795**	0.895	**0.875**
mBERT	0.841	0.812	0.807	0.773	0.888	0.863

may provide better results than pre-training a multi-dialectal model on considerably larger data. In contrast to MARBERT, which underwent training on a substantially larger corpus encompassing diverse Arabic dialects, DziriBERT exhibited consistent superiority in performance. In the final phase of evaluating writing styles (Code-switching), DziriBERT once again demonstrated its excellence, achieved an impressive F1 score of 87.5%. This exceptional performance is attributed to DziriBERT's training on a relatively modest yet substantial corpus of Algerian text, comprising both Arabic and Latin characters.

As seen in Table 7, our models performed the best in terms of accuracy on both Dz and Arabizi writing styles. In this comparison, we compared our models' results to those of (Touileb and Barnes, 2021), who fine-tuned the multilingual BERT on their own data, (Seddah et al., 2020), who used a feature-based alVWTagger, and (Muller et al., 2020), who use mBERT and the StanfordNLP tagger.

5.2.2 Experimental series 2

The experiments are the same as in the previous series, except this time we train all the models on the MSA dataset and test them on the Algerian dataset with the three writing styles. Table 8 shows the accuracy and F1 score results for each model on the three writing styles.

Table 8 demonstrated that the models performed poorly on the three writing styles of the Algerian dialect as compared to the results obtained when the models were trained on the Algerian dataset. With 43%, 20%, and 49.1% F1 scores in Dz, Arabizi, and code-switched, respectively, DziriBERT and MARBERT outperformed the other models in the three writing styles. We may support this with the pre-training corpus for both models, which are trained only on Arabic dialects for MARBERT and Algerian dialects for DziriBERT. Furthermore, as compared to the Arabizi style, both models behaved well in the Dz and code-switched styles.

6 Conclusion

In this study, we assessed POS tagging for the Algerian dialect using transformer-based pre-trained

Table 7: Comparison of our best model with previous works in terms of accuracy.

Model	Dz	Arabizi
Touileb et al. (Touileb and Barnes, 2021)	82.5	76.3
Seddah et al. (Seddah et al., 2020)	80.4	-
Muller et al. (Muller et al., 2020)	81.6	-
Our model	**0.871**	**0.831**

Table 8: Performance results on Algerian test set with the use of MSA dataset as a training set.

	Dz		**Arabizi**		**Code-switching**	
Model	**Accuracy**	**F1 score**	**Accuracy**	**F1 score**	**Accuracy**	**F1 score**
AraBERT base v0.2	0.443	0.409	0.199	0.101	0.515	0.462
AraBERT v02-twitter	0.444	0.410	0.171	0.107	0.521	0.471
MARBERT	0.450	0.417	0.226	**0.200**	0.540	0.487
DziriBERT	0.462	**0.430**	0.223	0.196	0.538	**0.491**
mBERT	0.437	0.408	0.181	0.152	0.504	0.464

models. Our research was organized to evaluate these models' performance across diverse writing styles of the Algerian dialect, with the primary objective of ascertaining how effectively models trained on MSA text can deal with the Arabic dialects. As a results, DziriBERT consistently achieved the highest F1 scores across all the writing styles, showcasing its adaptability and robustness in handling these variations of the Algerian dialect. Our model outperformed previous works in accuracy for Dz and Arabizi styles. Moreover, when models trained on MSA were tested on Algerian data, performance dipped, but DziriBERT and MARBERT maintained strong results, especially in Dz and code-switched styles due to the amount of Algerian dialect data seen during the pre-training phase. Overall, this highlights the importance of tailoring models to dialects due to significant differences. DziriBERT excelled, even with a small training corpus, offering promise for dialect-specific language tasks. Future research will explore POS tagging's impact in other tasks like named entity recognition, machine translation, and segmentation.

References

Amine Abdaoui, Mohamed Berrimi, Mourad Oussalah, and Abdelouahab Moussaoui. 2022. Dziribert: a pre-trained language model for the algerian dialect. arXiv:2109.12346v3.

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. 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 7088–7105, Online. Association for Computational Linguistics.

Karim Al-Haff, Mustafa Jarrar, Tymaa Hammouda, and Fadi Zaraket. 2022. Curras + baladi: Towards a Levantine corpus. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 769–778, Marseille, France. European Language Resources Association.

Rania Al-Sabbagh and Roxana Girju. 2012a. A supervised POS tagger for written Arabic social networking corpora. In *Proceedings of KONVENS 2012*, pages 39–52. ÖGAI.

Rania Al-Sabbagh and Roxana Girju. 2012b. YADAC: Yet another dialectal Arabic corpus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2882–2889, Istanbul, Turkey. European Language Resources Association (ELRA).

Faisal Al-Shargi, Aidan Kaplan, Ramy Eskander, Nizar Habash, and Owen Rambow. 2016. Morphologically annotated corpora and morphological analyzers for Moroccan and sanaani yemeni Arabic. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1300–1306, Portorož, Slovenia. European Language Resources Association (ELRA).

Mohammed Albared, Nazlia Omar, and Mohd. Juzaiddin Ab Aziz. 2011. Developing a competitive hmm arabic pos tagger using small training corpora. In *Intelligent Information and Database Systems*, pages 288–296, Berlin, Heidelberg. Springer Berlin Heidelberg.

Randah Alharbi, Walid Magdy, Kareem Darwish, Ahmed AbdelAli, and Hamdy Mubarak. 2018. Part-of-speech tagging for Arabic Gulf dialect using Bi-LSTM. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Wasan AlKhwiter and Nora Al-Twairesh. 2021. Part-of-speech tagging for arabic tweets using crf and bi-lstm. *Computer Speech Language*, 65:101138.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Laith H Baniata, Seyoung Park, and Seong-Bae Park. 2018. A multitask-based neural machine translation model with part-of-speech tags integration for arabic dialects. *Applied Sciences*, 8(12):2502.

Rahma Boujelbane, Mariem Ellouze, Frédéric Béchet, and Lamia Belguith. 2014. De l'arabe standard vers l'arabe dialectal : projection de corpus et ressources linguistiques en vue du traitement automatique de l'oral dans les médias tunisiens [from Modern Standard Arabic to Tunisian dialect: corpus projection and linguistic resources towards the automatic processing of speech in the Tunisian media]. *Traitement Automatique des Langues*, 55(2):73–96.

Eric Brill. 1994. Some advances in transformation-based part of speech tagging. arXiv:cmp-lg/9406010.

Kareem Darwish, Mohammed Attia, Hamdy Mubarak, Younes Samih, Ahmed Abdelali, Lluís Màrquez, Mohamed Eldesouki, and Laura Kallmeyer. 2020. Effective multi-dialectal arabic pos tagging. *Natural Language Engineering*, 26(6):677–690.

Kareem Darwish, Hamdy Mubarak, Ahmed Abdelali, and Mohamed Eldesouki. 2017. Arabic POS tagging: Don't abandon feature engineering just yet. In *Proceedings of the Third Arabic Natural Language Processing Workshop*, pages 130–137, Valencia, Spain. Association for Computational Linguistics.

Kareem Darwish, Hamdy Mubarak, Ahmed Abdelali, Mohamed Eldesouki, Younes Samih, Randah Alharbi, Mohammed Attia, Walid Magdy, and Laura Kallmeyer. 2018. Multi-dialect Arabic POS tagging: A CRF approach. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Mona Diab. 2009. Second generation amira tools for arabic processing: Fast and robust tokenization, pos tagging, and base phrase chunking. In *2nd international conference on Arabic language resources and tools*, volume 110, page 198.

Kevin Duh and Katrin Kirchhoff. 2005. POS tagging of dialectal Arabic: A minimally supervised approach. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 55–62, Ann Arbor, Michigan. Association for Computational Linguistics.

Amany Fashwan and Sameh Alansary. 2022. Developing a tag-set and extracting the morphological lexicons to build a morphological analyzer for Egyptian Arabic. In *Proceedings of the The Seventh Arabic Natural Language Processing Workshop (WANLP)*, pages 142–160, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Nizar Habash. 2010. *Introduction to Arabic Natural Language Processing*. Synthesis Lectures on Human Language Technologies. Morgan Claypool Publishers.

Nizar Habash and Owen Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580, Ann Arbor, Michigan. Association for Computational Linguistics.

Nizar Habash, Owen Rambow, and Ryan Roth. 2009. Mada+ tokan: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In *Proceedings of the 2nd international conference on Arabic language resources and tools (MEDAR), Cairo, Egypt*, volume 41, page 62.

Ahmed Hamdi, Alexis Nasr, Nizar Habash, and Núria Gala. 2015. POS-tagging of Tunisian dialect using Standard Arabic resources and tools. In *Proceedings of the Second Workshop on Arabic Natural Language Processing*, pages 59–68, Beijing, China. Association for Computational Linguistics.

Go Inoue, Salam Khalifa, and Nizar Habash. 2022. Morphosyntactic tagging with pre-trained language models for Arabic and its dialects. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1708–1719, Dublin, Ireland. Association for Computational Linguistics.

Mustafa Jarrar, Nizar Habash, Diyam Akra, and Nasser Zalmout. 2014. Building a corpus for palestinian Arabic: a preliminary study. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 18–27, Doha, Qatar. Association for Computational Linguistics.

Graham Katz and Mona Diab. 2011. Introduction to the special issue on arabic computational linguistics. *ACM Transactions on Asian Language Information Processing*, 10(1).

Salam Khalifa, Nizar Habash, Dana Abdulrahim, and Sara Hassan. 2016. A large scale corpus of Gulf Arabic. In *Proceedings of the Tenth International*

Conference on Language Resources and Evaluation (LREC'16), pages 4282–4289, Portorož, Slovenia. European Language Resources Association (ELRA).

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.

Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramón Fermandez, Silvio Amir, Luís Marujo, and Tiago Luís. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon, Portugal. Association for Computational Linguistics.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, Mona Diab, Nizar Habash, Owen Rambow, and Dalila Tabessi. 2006. Developing and using a pilot dialectal Arabic treebank. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Benjamin Muller, Benoit Sagot, and Djamé Seddah. 2020. Can multilingual language models transfer to an unseen dialect? a case study on north african arabizi. *arXiv preprint arXiv:2005.00318*.

Abdul Munem Nerabie, Manar AlKhatib, Sujith Samuel Mathew, May El Barachi, and Farhad Oroumchian. 2021. The impact of arabic part of speech tagging on sentiment analysis: A new corpus and deep learning approach. *Procedia Computer Science*, 184:148–155. The 12th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 4th International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1094–1101, Reykjavik, Iceland. European Language Resources Association (ELRA).

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Wael Salloum and Nizar Habash. 2011. Dialectal to Standard Arabic paraphrasing to improve Arabic-English statistical machine translation. In *Proceedings of the First Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties*, pages 10–21, Edinburgh, Scotland. Association for Computational Linguistics.

Younes Samih, Mohamed Eldesouki, Mohammed Attia, Kareem Darwish, Ahmed Abdelali, Hamdy Mubarak, and Laura Kallmeyer. 2017. Learning from relatives: Unified dialectal Arabic segmentation. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 432–441, Vancouver, Canada. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Djamé Seddah, Farah Essaidi, Amal Fethi, Matthieu Futeral, Benjamin Muller, Pedro Javier Ortiz Suárez, Benoît Sagot, and Abhishek Srivastava. 2020. Building a user-generated content North-African Arabizi treebank: Tackling hell. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1139–1150, Online. Association for Computational Linguistics.

Samia Touileb and Jeremy Barnes. 2021. The interplay between language similarity and script on a novel multi-layer Algerian dialect corpus. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3700–3712, Online. Association for Computational Linguistics.

Kristina Toutanvoa and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 63–70, Hong Kong, China. Association for Computational Linguistics.

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

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.

Daniel Zeman, Joakim Nivre, Mitchell Abrams, Elia Ackermann, Noëmi Aepli, Željko Agic, Lars Ahrenberg, Chika Kennedy Ajede, Gabriele Aleksandraviciute, Lene Antonsen, et al. 2020. Universal dependencies 2.6. *LINDAT/CLARIAHCZ digital library at the Institute of Formal and Applied Linguistics (UFAL), Faculty of Mathematics and Physics, Charles University. url: http://hdl. handle. net/11234/1-3226*.

A Neural Network Approach to Ellipsis Detection in Ancient Greek

Giuseppe G. A. Celano
Leipzig University
Faculty of Mathematics and Computer Science
Institute of Computer Science
celano@informatik.uni-leipzig.de

Abstract

In the present article, five neural networks models for prediction of the number of elliptical nodes in Ancient Greek sentences are compared. The models are trained on dependency treebank data, where elliptical nodes are introduced if and only if they govern nodes that would otherwise become orphans. As exact word forms of elliptical nodes cannot often be identified (and therefore be annotated) in Ancient Greek, the task is modeled as a multiclass classification one, where each sentence is associated with zero, one, two, or more than two elliptical nodes. The study shows that pretrained BERT token embeddings allow achievement of the best performance. A model, which is the first of its kind, is made available for further research.

1 Introduction

In linguistics, “ellipsis” can be broadly defined as the phenomenon whereby a sentence lacks one or more constituents that are left implied, but can be inferred from the linguistic context.

Depending on the language analyzed and the theoretical framework, different descriptions and definitions of ellipsis have been proposed in the theoretical linguistics literature (see, for example, Van Craenenbroeck and Temmerman, 2019 for a general overview).

Ellipsis in Ancient Greek has often been associated with, or treated as, a stylistic device in the older literature (see, for example, Kühner et al., 1965, pp. 558–571, who provide a long list of examples, and Schwyzer, 1971, pp. 707–710). In the more recent literature, however, the phenomenon has been investigated in word order studies. Gaeta and Luraghi (2001), for example, distinguish three different types of ellipsis: gapping, split coordination, and coordination reduction. Gapping occurs when there are at least two contrasted constituents:

(1) ὥσπερ Ἐμπεδοκλῆς φησὶ$_i$ φιλίαν, ἄλλος δέ τις Ø$_i$ πῦρ, ὁ δὲ Ø$_i$ ὕδωρ ἢ ἀέρα
‘as Empedocles holds of Love, another thinker of fire, and another of water or air’[1] (Arist. Metaph. 996a 8)

In Example (1), the contrasted constituents are, on the one hand, “Ἐμπεδοκλῆς,” “ἄλλος (δέ) τις,” and “ὁ (δὲ),” and, on the other, “φιλίαν,” “πῦρ,” and “ὕδωρ ἢ ἀέρα,” the elliptical constituent being the verb “φησὶ.” This sentence is an example of rightward gapping, because the elliptical verbs refer back to “φησὶ.” This is not the only type of gapping in Ancient Greek, in that an elliptical constituent could also refer to a following constituent (leftward gapping).

Examples of split coordination and coordination reduction are given by Examples (2) and (3) , respectively:

(2) ἔπεσθαι$_i$ δέ οἱ$_z$ τῶν μαχίμων μὲν οὐδένα ἀνδρῶν, καπήλους δὲ καὶ χειρώνακτας καὶ ἀγοραίους ἀνθρώπους Ø$_{iz}$
‘and none of the warriors would go with him, but only merchants and craftsmen and traders’ (Hdt. 2.141.4)

(3) ἱστία$_i$ μὲν στείλαντο, θέσαν δ᾽ Ø$_i$ ἐν νηῒ μελαίνῃ
‘they furled the sail, and stowed it in the black ship’ (Hom. Il. 1.433)

In Example (2), the verb and its second argument “οἱ” are omitted, while in Example (3) only the object is.[2] Ellipsis of direct object (and some other second arguments)[3] often occurs also in complex

[1] Translations in the article derive from the Perseus Digital Library at http://www.perseus.tufts.edu/hopper/.

[2] As Ancient Greek has a rather free, information structure-based word order, indication of the position of the ellipsis in the examples is to be considered approximate.

[3] I leave aside the question of the relationship between ellipsis and verb valency in Ancient Greek, which has not yet been investigated satisfactorily. Indeed, in some examples,

sentences, as in Example (4) (Luraghi, 2003, p. 170) :

(4) ὁ δὲ ἐμπιμπλὰς ἁπάντων$_i$ τὴν γνώμην, ἀπέπεμπε Ø$_i$
'having satisfied the expectation of all, he dismissed them' (Xen. Anab. 1.7.8)

The variety of ellipsis types, of which Example (1)–(4) just offer a meager glimpse, is clearly dependent on information structure, which has been proved to determine the high configurational complexity of Ancient Greek word order (Celano, 2013; Dick, 1995). As can be expected, therefore, it is not only challenging to describe and explain ellipsis in Ancient Greek, but also to annotate it.

In the following sections, I present five models built to predict the number of elliptical nodes in Ancient Greek sentences. More precisely, an overview of related work is given in Section 2. Section 3 provides details on the data used for the present study, while Section 4 describes ellipsis annotation. Five models to predict ellipsis are compared in Section 5. A few concluding remarks are contained in Section 6.

2 Related Work

Most of the previous ellipsis-related work conducted in computational linguistics/NLP has so far focused on verb phrase ellipsis (VPE) in English.[4]

Hardt (1997) describes a rule-based system to resolve ellipsis in 644 examples from the Penn Treebank and, more recently, Hardt (2023) tests the ability of a number of Large Language Models to understand ellipsis. Nielsen (2004) tests a variety of machine learning algorithms for VPE detection using the British National Corpus and the Penn Treebank. Bos and Spenader (2011) provide an account for the creation of a new VPE corpus, by detailing the annotation process of the 25 sections of the Wall Street Journal contained in the Penn Treebank. Bos and Spenader's (2011) corpus has also been used by Zhang et al. (2019), which seems to be the first neural networks-based study on VPE.

In the above-mentioned literature, VPE processing for English is divided into two main related tasks: (i) VPE detection and (ii) VPE resolution.

Codepoint	F	RF
Greek Coronis (U+1FBD)	7,224	0.16
Combining Comma Above (U+0313)	28,581	0.63
Apostrophe (U+0027)	11	0.00
Right Single Quotation Mark (U+2019)	4,481	0.10
Modifier Letter Apostrophe (U+02BC)	5,269	0.12
	45,566	1

Table 1: Unicode characters used to encode the apostrophe in the original treebank data with their (relative) frequencies.

VPE detection is modeled as a binary classification task outputting whether or not auxiliaries, such as "do," "be," or "have," are used as triggers, i.e., they replace a preceding VP. On the other hand, VPE resolution aims to identify the antecedent a given trigger refers to: more precisely, the task involves identification of candidate antecedents, over each of which a binary classification task is performed (Zhang et al., 2019).

As will be shown in the following sections, the task at hand to predict ellipsis in Ancient Greek differs from the above-mentioned studies because it only concerns detection of the number of elliptical nodes in a sentence, without identification of its word form or resolution. This task indeed depends on the nature of the Ancient Greek language, where, typically, there is no trigger constituent for an elliptical node, and its exact form and position are often unclear.

3 The Data

There exist two major related data sets containing morphosyntactic annotations of Ancient Greek texts, which also include annotation for ellipsis: the Ancient Greek Dependency Treebank[5] and the Dependency Treebanks of Ancient Greek Prose (Gorman, 2020).[6] [7]

The two treebanks, which have been annotated using the same annotation scheme, have been merged together (for convenience, I henceforth refer to this data set as "Ancient Greek Dependency Treebank"): the data set comprises 187 files,

one might posit one-argument verbs instead of two-argument verbs with ellipsis of an object.

[4] Noun ellipsis detection, which is less relevant for the present study, has recently been investigated by Khullar (2020).

[5] https://github.com/PerseusDL/treebank_data/releases/tag/v2.1_IGDS.

[6] I downloaded the data from the main branch of https://github.com/vgorman1/Greek-Dependency-Trees, which contains more recent data than the released one at https://zenodo.org/record/3596076#.XlZ7CxP7Su4.

[7] I limited the study to the above-mentioned data sets. There exist, however, a few others: in particular, Pedalion, which is available in a beta version, is worthy of note (Keersmaekers et al., 2019).

Model	Class Weights	Accuracy	Precision		Recall		F1	
			M	w	M	w	M	w
Feedforward$_{Baseline}$		0.77	0.42	0.70	0.27	0.77	0.27	0.71
	✓	0.67***	0.33	0.70	0.34	0.67	0.33	0.68
Feedforward$_{DBBE-BERT}$		**0.85*** **	**0.64**	**0.84**	0.50	**0.85**	**0.55**	**0.84**
	✓	0.80***	0.58	0.81	**0.53**	0.80	0.52	0.80
Transformer$_{DBBE-BERT}$		**0.85*** **	0.61	0.83	0.50	**0.85**	0.54	**0.84**
	✓	0.79***	0.54	0.81	**0.53**	0.79	0.51	0.79
Transformer$_{WordPiece35000}$		0.78**	0.44	0.71	0.28	0.78	0.29	0.71
	✓	0.54***	0.31	0.69	0.39	0.54	0.26	0.59
LSTM$_{DBBE-BERT}$		**0.85*** **	0.61	0.83	0.49	**0.85**	0.53	0.83
	✓	0.81***	0.57	0.81	**0.53**	0.81	0.53	0.81

Table 2: Model metrics calculated on the test set (M = macro, w = weighted). Statistical differences from Feedforward$_{Baseline}$ are calculated using Stuart-Maxwell tests and reported in the accuracy column ($p < 0.05$ (*), $p < 0.01$ (**), and $p < 0.001$ (***)). The highest values are indicated in bold.

amounting to 54,925 sentences and 1,063,984 tokens.[8] In order to facilitate further processing, the data set has been normalized with respect to (i) Unicode form and characters and (ii) tokenization scheme.

The texts have been NFC normalized, and there has been an attempt to make the encoding of the apostrophe uniform. There are at least five different Unicode characters with indistinguishable glyphs that are interchangeably used in the treebank texts (see Table 1): they have all been converted into Modifier Letter Apostrophe (U+02BC).

There has also been an attempt to normalize the data with respect to the tokenization scheme: indeed, while most Ancient Greek graphic words coincide with morphosyntactic words (and therefore with the tokens found in the Ancient Greek Dependency Treebank), there are two main cases where this does not hold true: (i) (negative) conjunctions and (ii) words contracted by crasis.

Conjunctions include examples such as “οὐδὲ” (“and/but not”), which is split into “οὐ” (“not”) and “δὲ” (“and/but”): there are 12 such conjunctions[9] and, in the final data set, they have all been segmented.

The words contracted by crasis are words that are univerbated for phonological reasons: an example is “κἀγὼ,” which consists of the word “καὶ” (“and”) and “ἐγὼ” (“I”). In the original treebank data, about half of all crases are split: it has been found heuristically that 1,371 cases of crasis are split, while 1,110 are not.[10] Since identification, segmentation, and morphosyntactic analysis of crases is challenging, they have not been modified in the final data set. The data and the best model (Feedforward-DBBE-BERT) are made available online[11] for further research.

4 Ellipsis annotation

There are two main challenges with reference to ellipsis annotation in treebanks: (i) identification of ellipsis and (ii) its formal representation.

In Ancient Greek, ellipsis is typically not signaled by a trigger such as auxiliaries in English. On the contrary, its presence can be inferred from linguistic context, as the following example shows:

(5) κρατουμένων μὲν γὰρ ἐπίστασθε ὅτι πάντα ἀλλότρια
‘for when men are conquered, you are aware that all their possessions become the property of others’ (Xen. Anab. 3.2.28)

In Example (5), the object clause can be properly annotated only by positing existence of an elliptical verb connecting “πάντα” and “ἀλλότρια,” as Figure 1 shows. Indeed, there is only one annotation rule for ellipsis annotation in the Ancient Greek Dependency Treebank: an elliptical node is recog-

[8] The data set is made available at https://git.informatik.uni-leipzig.de/celano/ellipsis_Ancient_Greek.

[9] The full list is available at https://git.informatik.uni-leipzig.de/celano/ancientgreeknlp/-/blob/master/tokenize/texts/to-tokenize.xml.

[10] According to this calculation, crases would amount to about 0.23% of all treebank tokens.

[11] https://git.informatik.uni-leipzig.de/celano/ellipsis_Ancient_Greek.

nized if and only if it is necessary to build a correct syntactic tree, i.e., the posited elliptical node has syntactic dependents.

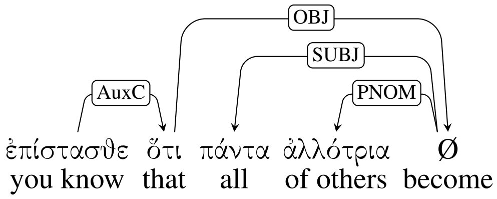

OBJ
SUBJ
AuxC
PNOM
ἐπίστασθε ὅτι πάντα ἀλλότρια Ø
you know that all of others become

Figure 1: (Partial) linguistic tree for Xen. Anab. 3.2.28.

If an elliptical node, as in Example (5), meets this condition, a new token is added at the end of the relevant sentence: such formalization has the advantage of allowing for the elliptical node to function as any other node, and therefore receive annotation for its head and its syntactic function. A disadvantage of this annotation is however that the number of original sentence tokens changes unpredictably, and therefore comparison of different annotations, as well as further machine learning processing, presents an added layer of complexity.[12]

In Example (5), the elliptical verb is likely to be "γίγνομαι," to be added after "ἀλλότρια": however, both form and position of an elliptical constituent are often ambiguous in practice. For this reason, ellipsis representation has been normalized in the data set, so that its form always corresponds, conventionally, to a number in squared brackets (e.g., `[0]`)[13], and its position is always at the end of a sentence, after any other non-elliptical node.

5 Experiment

Prediction of the number of elliptical nodes in Ancient Greek sentences has been modeled as a multiclass classification task, with 4 class labels for (i) none, (ii) 1, (iii) 2, and (iv) 3 or more elliptical nodes per sentence, respectively.

As Figure 2 shows, sentences with no ellipsis and up to 3 elliptical nodes represent about 99.5% of all sentences: for better model performance, therefore, rare sentences with more than 3 elliptical nodes have been included in the class representing 3 elliptical nodes.

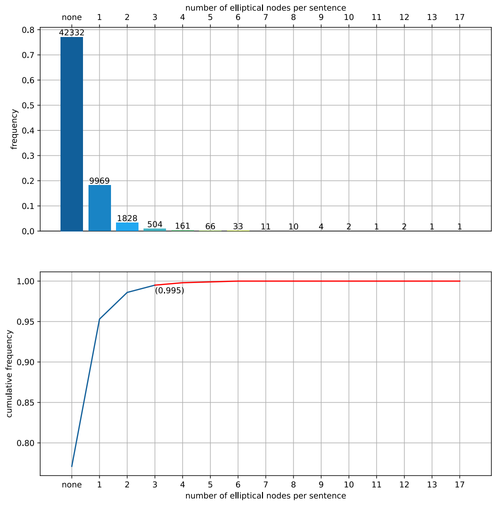


Figure 2: Key statistics for elliptical nodes in the whole data set.

In the following sections, I compare five machine learning models. Each of them has also been trained with class weights[14] because of the class-unbalanced data set. The data set has been divided into training (∼80%), development (∼10%), and test (∼10%) data sets.[15] Accuracy, macro- and weighted-average precision, recall, and F1 are the metrics used for evaluation. To evaluate statistical significance, Stuart-Maxwell marginal homogeneity tests are used.

5.1 Model Architectures

A feedforward neural network has been chosen as a baseline model (Feedforward-Baseline), and its output has been compared to those of four different models: (i) a feedforward model with pretrained BERT token embeddings (Feedforward-DBBE-BERT); (ii) an encoder-only transformer with pretrained BERT token embeddings (Transformer-DBBE-BERT); (iii) an encoder-only transformer with randomly initialized token embeddings (Transformer-WordPiece35000); (iv) an LSTM neural network with pretrained BERT

[12] This is probably the reason why, in Universal Dependencies, ellipsis is annotated at the level of syntactic label (`https://universaldependencies.org/u/overview/specific-syntax.html#ellipsis`): this solution, however, suffers from the disadvantage of rendering syntactic annotation rather obscure.

[13] `[0]` means one elliptical node, `[1]` two elliptical nodes, and so on and so forth.

[14] Weights are calculated using the method `compute_class_weight` of `sklearn.utils.class_weight` (with `balanced` argument).

[15] Because of unbalancedness, the development and test data sets have been selected through stratified sampling, with *strata* being the classes of the dependent variable.

token embeddings (LSTM-DBBE-BERT).[16]

As Figure 3 shows, the baseline model consists of 3 ReLU-activated linear layers (with 5,000, 2,000, and 1,000 units, respectively), a 0.2 dropout layer, and a final linear layer with softmax activation. Sentences are vectorized using TF-IDF scores calculated using the sentences themselves as documents.

An already existing BERT model (DBBE-BERT) (Singh et al., 2021)[17] has been fine-tuned for the Feedforward-DBBE-BERT model. The DBBE-BERT model is a Masked Language Model that has been trained on Modern and Ancient Greek data. The tokenizer vocabulary of the model consists of 35,000 tokens and each token embedding has 768 dimensions. The DBBE-BERT token embeddings have been fed into a global average pooling layer followed by a 0.2 dropout layer, two linear layers with 500 and 100 units respectively, a 0.2 dropout layer, and the final softmax-activated linear layer outputting probabilities (the model architecture is the same as that of the LSTM model shown in Figure 6, except for the LSTM layer, which is replaced by a global average pooling layer).

Since sentences are composed of words, an encoder-only transformer has been tested to leverage relationships between them. Two variants of the encoder-only transformer have been tested.

The first variant (see Figure 4) aims to test whether a transformer layer can improve the performance of the DBBE-BERT token embeddings, which also rely on a transformer-based architecture.

The DBBE-BERT token embeddings are fed into an encoder-only transformer based on Vaswani et al. (2017), mainly consisting of a 2-head attention layer and a feedforward network with two 768-unit linear layers. The output of the transformer is then fed into a further feedforward component, with two linear layers with 500 and 100 units, respectively. All dropout and layer normalization layers have arguments 0.2 (rate) and 1e-6 (epsilon), respectively.

The second variant of the encoder-only transformer aims to assess the contribution of pretrained DBBE-BERT token embeddings. The model architecture is the same as the one described above (see Figure 4), except for token embeddings, which are not pretrained, but randomly initialized.

The tokens for the latter model are identified by a WordPiece algorithm with a vocabulary of 35,000 based on the texts of *Opera Graeca Adnotata* (Celano, 2023), a 34,172,140 token standoff annotation corpus. The texts in *Opera Graeca Adnotata* come from the Perseus Digital Library and First1KGreek projects,[18] and therefore coincide, for the most part, with those used to calculate the DBBE-BERT token embeddings. Each randomly initialized token embedding has 2,000 dimensions, with the transformer encoder's feedforward network consisting of two linear layers with 2,000 (and not 768) units each.

The last tested model is a LSTM neural network (see Figure 6), with a LSTM layer of 1,000 units, followed by two linear layers with 500 and 100 units, respectively, and ReLU activation (the rate of both dropout layers is 0.2). Token embeddings are calculated by the same pretrained DBBE-BERT model used for the Transformer-DBBE-BERT and Feedforward-DBBE-BERT models.

All models have been trained with the Adam optimizer with a learning rate of 1e-6. Early stopping has been determined by monitoring validation loss with a patience of 2 epochs.

5.2 Results

As Table 2 shows, the baseline model without class weights seems to achieve a good accuracy score: however, this score is misleading, in that it is the same as that of a dummy classifier always predicting the most frequent class label (i.e., the `none` label, which means absence of elliptical nodes).

Notably, Transformer-WordPiece35000 without class weights provides results very similar to the baseline ones (see also Figure 5 and 7). The model performance turns out to be statistically different from the baseline's one according to a Stuart-Maxwell test, whose probability value is greater than 0.001, but lower than 0.01. The feedforward model, the transformer, and the LSTM model trained with the DBBE-BERT token embeddings (without class weights) show the best results, with accuracy scores that are 8% higher than the baseline's one.[19]

[16] These token embeddings, as emerges in the following paragraphs, are the DBBE-BERT ones.

[17] The model is called by the authors "Extended Ancient Greek BERT" and, in this paper, "DBBE-BERT" for brevity's sake (DBBE is the acronym of the project "Database of Byzantine Book Epigrams," within which the model was developed).

[18] https://github.com/PerseusDL/canonical-greekLit; https://github.com/OpenGreekAndLatin/First1KGreek.

[19] The Feedforward-DBBE-BERT model is made available

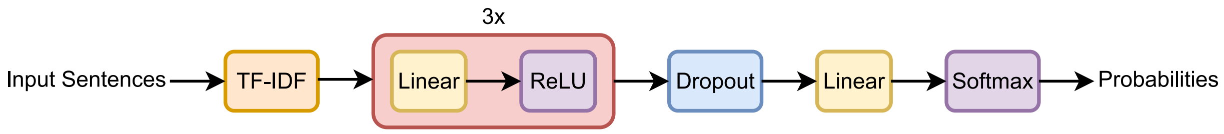


Figure 3: Baseline model architecture.

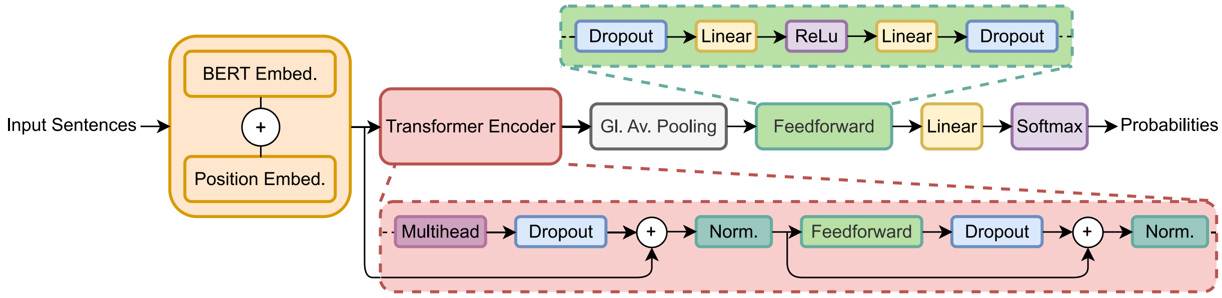


Figure 4: Encoder-only transformer architecture.

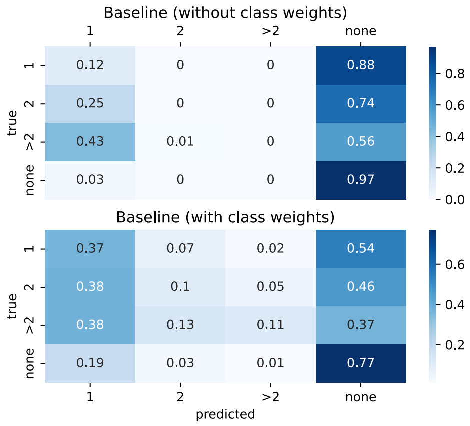


Figure 5: Confusion matrix for the baseline model showing false negative rates and recall (test data set).

The performances of models with class weights are worse than those of the corresponding models without weigths, and the their differences are statistically significant ($p < 0.001$).

Feedforward-DBBE-BERT's, Transformer-DBBE-BERT's, and LSTM-DBBE-BERT's performances are comparable: however, when their outputs are tested with Stuart-Maxwell tests (pairwise), only Feedforward-DBBE-BERT's and LSTM-DBBE-BERT's results turn out to be not statistically different ($p > 0.05$).

The results suggest that the pretrained DBBE-BERT token embeddings play a crucial role. The transformer architecture of Transformer-WordPiece35000 guarantees a context-aware token representation, but this seems to be not enough for the task at hand, if token embeddings are randomly initialized.

5.3 Error Analysis

As Figure 5 shows, the baseline model without class weights can almost exclusively classify sentences as belonging to class none and 1. Most sentences of class none are classified correctly (recall 0.97), even if the classifier also tends to incorrectly label as none most sentences with classes 1, 2, and >2. 12% of the sentences with label 1 are classified correctly, but most of the sentences the classifier labels as 1 are misclassified (see also Table 3). The baseline model with class weights identifies more sentences of class 2 and >2, but precision and recall scores for these classes are very low (0.08, 0.1 and 0.13 and 0.11, respectively), and, more in general, its overall performance, as shown by Table 2, is worse than that of the baseline model without weights.

Figure 7 shows that the confusion matrix for Transformer-WordPiece35000 is surprisingly comparable to the baseline's one (without weights), in that there are almost no predicted sentences of class 2 or >2, and the classifiers' scores for class 1 and none are also very similar.

Analysis of the confusion matrices for Feedforward-DBBE-BERT, Transformer-DBBE-BERT, and LSTM-DBBE-BERT in Figure 7 reveals that they are very similar. Most sentences with no or one elliptical node are correctly classified. Classification of sentences with 2 or more than 2 elliptical nodes remains a challenge, since most of them are misclassified. There is however an improvement in comparison to

at https://git.informatik.uni-leipzig.de/celano/ellipsis_Ancient_Greek.

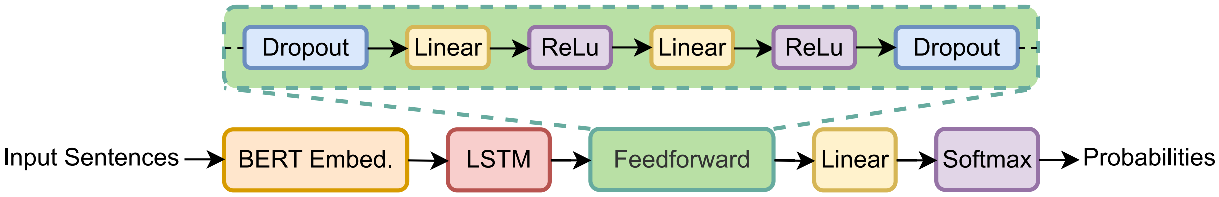


Figure 6: LSTM architecture.

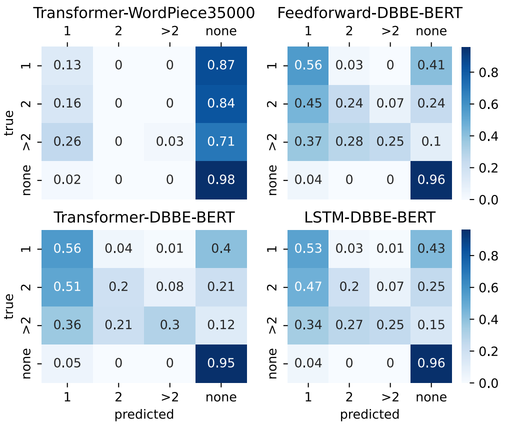


Figure 7: Confusion matrices for the models (without class weights) showing false negative rates and recall (test data set).

the baseline model: Feedforward-DBBE-BERT, for example, correctly classify 24% of 2-class sentences and 25% of >2-class sentences. The improvement of the models trained with the pretrained DBBE-BERT token embeddings is also confirmed by the precision, recall, and F1 scores per class reported in Table 3. I hypothesize that the bad performance in identifying sentences with two or more than two elliptical nodes much depends on the data set being highly unbalanced.

6 Conclusion

The present study compared five neural network models for prediction of the number of elliptical nodes in Ancient Greek sentences. The comparison showed that models with pretrained BERT token embeddings fine-tuned for the task at hand achieved the best results, with a good accuracy score (0.84), but a not very high macro-averaged F1 score (0.55 for Feedforward-DBBE-BERT). In comparison, the transformer architecture with randomly initialized token embeddings (Transformer-WordPiece35000) scored significantly worse, with a performance comparable to that of the baseline feedforward network with TF-IDF sentence vectorization (without class weights).

Model	Class	Precision	Recall	F1
BASELINE	1	0.38	0.12	0.18
	2	0.50	0.00	0.01
	>2	0	0	0
	none	0.80	0.97	0.88
FF-BERT	1	0.67	0.56	0.61
	2	0.47	0.24	0.32
	>2	0.52	0.25	0.34
	none	0.90	0.96	0.93
TR-BERT	1	0.64	0.56	0.59
	2	0.40	0.20	0.27
	>2	0.48	0.30	0.37
	none	0.90	0.95	0.93
LSTM-BERT	1	0.65	0.53	0.59
	2	0.43	0.20	0.28
	>2	0.46	0.25	0.32
	none	0.89	0.96	0.92

Table 3: Precision, recall, and F1 scores per class for the models Baseline, Feedforward-DBBE-BERT (FF-BERT), Transformer-DBBE-BERT (TR-BERT), and LSTM-DBBE-BERT (LSTM-BERT) (test data set).

The study also showed that performances of model architectures of different complexity fed with the same pretrained BERT token embeddings (i.e., Feedforward-DBBE-BERT, Transformer-DBBE-BERT, and LSTM-DBBE-BERT without class weights) proved to be comparable.

Limitations

Since annotation was performed by single annotators, some variability in ellipsis identification has to be expected in the original data.

Acknowledgements

This work has been supported by the German Research Foundation (DFG project number 408121292).

References

Johan Bos and Jennifer Spenader. 2011. An annotated corpus for the analysis of VP ellipsis. *Language Resources and Evaluation*, 45:463–494.

Giuseppe G. A. Celano. 2013. Argument-focus and predicate-focus structure in Ancient Greek: Word order and phonology. *Studies in Language*, 37(2):241–266.

Giuseppe G. A. Celano. 2023. *Opera Graeca Adnotata*. Version 0.1.0. Zenodo. https://doi.org/10.5281/zenodo.8158675.

Helma Dick. 1995. *Word Order in Ancient Greek: A Pragmatic Account of Word Order Variation in Herodotus*. J.C. Gieben, Amsterdam.

Livio Gaeta and Silvia Luraghi. 2001. Gapping in Classical Greek prose. *Studies in Language*, 25(1):89–113.

Vanessa B. Gorman. 2020. Dependency treebanks of Ancient Greek prose. *Journal of Open Humanities Data*, 6(1):1.

Daniel Hardt. 1997. An empirical approach to VP ellipsis. *Computational Linguistics*, 23(4):525–541.

Daniel Hardt. 2023. Ellipsis-dependent reasoning: a new challenge for large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 39–47, Toronto, Canada. Association for Computational Linguistics.

Alek Keersmaekers, Wouter Mercelis, Colin Swaelens, and Toon Van Hal. 2019. Creating, enriching and valorizing treebanks of Ancient Greek. In *Proceedings of the 18th International Workshop on Treebanks and Linguistic Theories (TLT, SyntaxFest 2019)*, pages 109–117, Paris, France. Association for Computational Linguistics.

Payal Khullar. 2020. Exploring statistical and neural models for noun ellipsis detection and resolution in English. 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 139–145, Suzhou, China. Association for Computational Linguistics.

Raphael Kühner, Friedrich Blass, Bernhard Gerth, and William M. Calder. 1965. *Ausführliche Grammatik der Griechischen Sprache*. Wissenschaftliche Buchgesellschaft, Darmstadt.

Silvia Luraghi. 2003. Definite referential null objects in Ancient Greek. *Indogermanische Forschungen*, 108:167–194.

Leif Arda Nielsen. 2004. Robust VPE detection using automatically parsed text. In *Proceedings of the ACL Student Research Workshop*, pages 49–54, Barcelona, Spain. Association for Computational Linguistics.

Eduard Schwyzer. 1971. *Griechische Grammatik*, volume 2. C.H. Beck'sche Verlagsbuchhandlung, München.

Pranaydeep Singh, Gorik Rutten, and Els Lefever. 2021. A pilot study for BERT language modelling and morphological analysis for ancient and medieval Greek. In *Proceedings of the 5th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 128–137, Punta Cana, Dominican Republic (online). Association for Computational Linguistics.

Jeroen Van Craenenbroeck and Tanja Temmerman. 2019. *The Oxford Handbook of Ellipsis*. Oxford University Press, Oxford.

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, pages 1–11.

Wei-Nan Zhang, Yue Zhang, Yuanxing Liu, Donglin Di, and Ting Liu. 2019. A neural network approach to verb phrase ellipsis resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7468–7475.

AraBERT and mBert: Insights from Psycholinguistic Diagnostics

Basma Sayah
Lab. d'Informatique & Mathématiques
Université Amar Telidji, Algeria
b.sayah@lagh-univ.dz

Attia Nehar
Lab. d'Informatique & Mathématiques
Ziane Achour University, Algeria
neharattia@univ-djelfa.dz

Hadda Cherroun
Lab. d'Informatique & Mathématiques
Université Amar Telidji, Algeria
hadda_cherroun@lagh-univ.dz

Slimane Bellaouar
Lab. Mathématiques & Sciences
University of Ghardaia, Algeria
bellaouar.slimane@univ-ghardaia.dz

Abstract

BERT, a groundbreaking large language model, has excelled in natural language processing tasks such as question answering. Motivated by a desire to understand BERT's knowledge and limitations across different languages, we build upon Alysson Ettinger's work by evaluating BERT Arabic versions using psycholinguistics. These diagnostics, designed to assess human brain linguistic abilities, cover aspects like common sense and pragmatic inference, which constitute fundamental knowledge for any pretrained language model. Upon translating these diagnostics into Arabic, the results of diagnostic assessments for mBERT in Arabic and AraBERT reveal linguistic deficiencies in mBERT and a moderate grasp in AraBERT. This emphasizes the need for further training on diverse texts, especially those related to everyday situations.

Keywords: AraBERT, mBert, Psycholinguistic, Linguistic evaluation, Arabic language models.

1 Introduction

Nowadays, large Language Models (LLMs) are the base of almost every Natural Language Processing (NLP) application. They are used in sentiment analysis (SA), question answering (QA), conversational agents, personal assistants, and robotics (et al., 2021).

Since the introduction of Transformers in 2017 (Vaswani et al., 2017), computers have demonstrated remarkable linguistic abilities, often comparable to those of humans. Consequently, a multitude of language models has emerged from the Transformer framework, addressing a variety of languages. Examples include: ELMo (Peters et al., 2018), BERT and mBERT (Devlin et al., 2019), GPT through all its versions (Radford et al., 2018), PaLM (et al., 2023).

Despite the popularity of these models and their impact across various fields, there is an urgent need for interdisciplinary efforts in order to understand the knowledge they infer and to discover their unknown failures. Previous studies have delved into various performance aspects, including task-specific evaluations (Jiang et al., 2021) (Wang et al., 2018), probing different layer (Conia and Navigli, 2022), and linguistics evaluations of humans on machines (Ettinger, 2020) (Lialin et al., 2022).

Unlike other languages, Arabic LLMs have not been extensively studied, despite some recent investigations (Albilali et al., 2021) (Abdelali et al., 2022). In this paper, we aim to fill this gap by investigating the capabilities of Arabic LLMs. Our initial step involves enhancing our understanding of what Arabic LLMs comprehend about the Arabic language by measuring their linguistic abilities through the discipline of psycholinguistics. Psycholinguistics, originally developed by linguists to assess the human brain's capacity to understand and produce language (Harley, 2013), serves as our guiding framework. Our investigation is specifically narrowed down to the Arabic language models araBERT and multilingual BERT

The rest of this paper is organized as follows. First, in Section 2, we introduce some preliminaries and concepts related to pre-trained LMs and psycholinguistic diagnostics. In Section 3, we review related literature that has considered the evaluation of LMs' linguistic abilities. The methodology of our investigation is presented in detail in Section 4. Finally, we report and discuss the results of the evaluation in Section˜ 5

2 Preliminaries

Driven by the purpose of this paper, this section offers a concise overview of Multilingual BERT and AraBERT. Subsequently, we delve into psycholinguistic aspects and psycholinguistic diagnostics that examine predictive human responses, all of which are relevant to the assessment of pre-trained Language Models.

2.1 Arabic LLMs and BERT

Arabic language models, especially those built on the BERT architecture, are pivotal in natural language processing. BERT, or Bidirectional Encoder Representations from Transformers (Devlin et al., 2019), constitutes a highly parallel deep neural network leveraging attention mechanisms for sequence prediction and generation (Vaswani et al., 2017).

Originally designed for language modeling and machine translation, transformers like BERT have evolved to handle more complex tasks, including computer vision (Nguyen et al., 2023). Specific BERT variations tailored for Arabic have been developed. Figure 1 provides a chronological overview of Arabic BERT models and other transformers pre-trained on diverse Arabic texts, encompassing dialects and Modern Standard Arabic (MSA) from platforms like social media, news, and academic content. For the purpose of this paper, we will focus specifically on two models: mBERT and AraBERT.

mBERT released by Devlin et al.,(2019) is a single-language model that was pre-trained using monolingual corpora in 104 languages, including Arabic. This enabled BERT to learn and generalize across multiple languages.

AraBERT developed by Antoun et al.,(2020) is a widely adopted model pre-trained on an extensive corpus of Modern Standard Arabic (MSA) texts. AraBERT is applied in various natural language processing (NLP) tasks, including text classification, named entity recognition (NER), and sentiment analysis (SA) in the Arabic language.

2.2 Psycholinguistics

Psycholinguistics, a subfield of linguistics, studies the mental processes involved in language acquisition, comprehension, and production (Harley, 2013). Within the domain of psycholinguistics, the study of human language processing incorporates fundamental metrics such as *Cloze probability* and *N400* amplitude (Kutas and Hillyard, 1984).

- Cloze probability is the likelihood or probability that individuals choose a specific word to complete a given context. It provides a quantifiable measure of how well a word fits into a particular linguistic context based on human responses.

- The N400 amplitude is a quantifiable electrical signal discerned in brain activity, particularly in electroencephalogram (EEG) recordings. The measurement of the N400 component's amplitude helps comprehend the brain's reaction to words that disrupt the contextual flow or are unexpected within a given sentence.

3 Related Work

In the literature, there is a growing effort to better understand the specific linguistic capacities achieved by neural Natural Language Processing (NLP) models. We have reviewed several studies that measured their performances and behaviors, categorizing them based on three criteria:

- Linguistic analysis: This category focuses on assessing the lexical, syntactic, and figurative skills of a language model.

- Tasks-based Analysis: This category involves evaluating the language model through specific tasks such as Sentiment Analysis (SA), Question Answering (QA), Translation, Named Entity Recognition (NER), and Dialect Identification.

- In-Depth Model Examination: This type of analysis delves into the inner workings of the model, considering aspects of explainability and probing.

In linguistic analysis, Ettinger, (2020) presents a set of diagnostics derived from human language experiments to systematically investigate the information utilized by language models during prediction generation in context. The study applies these diagnostics to assess the popular BERT model. The findings reveal that BERT demonstrates a general ability to distinguish between good and bad completions involving shared category or role reversal, though with less sensitivity compared to humans.

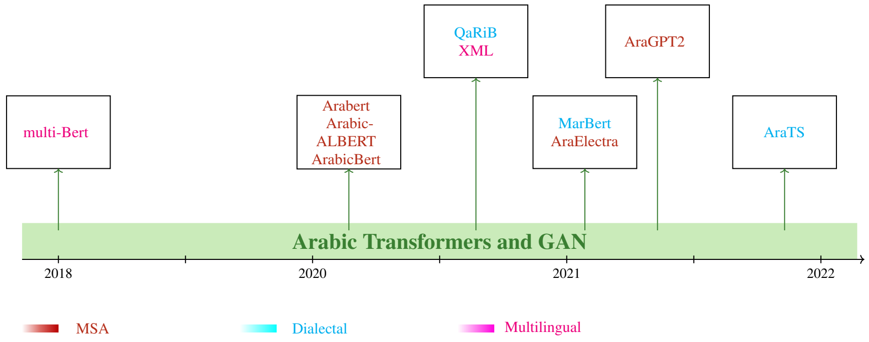


Figure 1: Some Arabic Transformers and GANs.

Additionally, BERT consistently retrieves noun hypernyms effectively. However, the model faces challenges in intricate tasks such as inference and role-based event prediction. Notably, BERT exhibits a clear insensitivity to the contextual impacts of negation.

In task-based analysis, Rönnqvist et al., () investigated mBERT's performance across languages and tasks. They found mBERT to be inferior to monolingual models, especially for Nordic languages. Chouikhi et al., (2021) addressed tokenization issues in Arabic Sentiment Analysis. Their approach, incorporating an Arabic BERT tokenizer instead of the basic BERT tokenizer, outperformed Arabic BERT and AraBERT models in classification quality and accuracy, particularly for dialect and MSA instances. Lialin et al., (2022) scrutinized 29 diverse model families, including T5, BART, and ALBERT, using the oLMpics benchmark and psycholinguistic probing datasets. Their study found that none of these models, when assessed in a zero-shot manner, could effectively address compositional questions, challenging the adequacy of current pre-training objectives for acquiring this skill.

In their in-depth model examination, Mickus et al., (2020) examined the semantic coherence of BERT's embedding space. They mention that, while showing a tendency towards coherence, BERT does not fully live up to the natural expectations for a semantic vector space. They discovered, in particular, that the position of a word in a sentence, despite having no meaning correlates, leaves an evident trace on the word embeddings and disrupts similarity relationships. Li et al., (2021) introduced a tool for probing surprisal at BERT's intermediate layers, employing density estimation with Gaussian models. They found a high correlation between surprisal and low token frequency in lower layers, decreasing in upper layers. Regarding morphosyntactic, semantic, and commonsense anomalies, the best-performing model (RoBERTa) exhibited surprisal in earlier layers for morphosyntactic anomalies, but not for semantic or commonsense anomalies. Abdelali et al., (2022) conducted a post-hoc examination of transformer models trained on diverse Arabic dialects. Using layer and neuron analysis, they found that word morphology is predominantly learned in lower and middle layers, syntactic dependencies are primarily captured in higher layers, and despite vocabulary overlap, models based on Modern Standard Arabic struggle to capture nuanced aspects of dialects. Neurons in embedding layers exhibit polysemous characteristics, while those in middle layers specialize in specific properties.

4 Methodology

In the assessment of the psycholinguistic skills of Arabic BERT models, we translated the psycholinguistic diagnostics from Ettinger's work into Arabic with the assistance of three Arabic native speakers, one of whom is a professional translator. Subsequently, we applied these diagnostics to AraBERTv2$_{base}$, AraBERTv2$_{large}$, and mBERT using the Python language in the Google Colab platform. Each diagnostic test involves sentences (contexts) with a missing word, and the task is to predict that missing word. Accurate predictions require the application of the targeted linguistic skills defined by these tests. The evaluation utilized the following metrics:

- **Word Prediction Accuracy** measures how often the language model correctly provides the expected item among its top k predictions and is designed to be the equivalent of Cloze probability in psycholinguistics (refer to Section 2).

- **Sensitivity Test** represents the percentage of items for which the probability assigned to a correct completion exceeds the probability assigned to the inappropriate one. This measure is designed to be the equivalent of the N400 in psycholinguistics (refer to Section 2).

- **Qualitative analysis** is the process of manually reviewing the results, making observations on the top k predictions, and understanding their relationships with each other and with the context, all in order to gain deeper insights into the skills of AraBERT.

All the diagnostic datasets and experiment code are shared and accessible on GitHub. [1] The following subsection provides a detailed description of the diagnostics employed in our evaluation.

4.1 CPRAG-102

This diagnostic is made up of 102 contexts. Each context comprises two consecutive sentences with a missing word (Federmeier and Kutas, 1999). In these contexts, predicting the missing word requires Common Sense to understand what is being described and Pragmatic Inference to understand how the second sentence relates to the first. Table 2 shows an example of CPRAG-102 and its Arabic translation. The 'Expected' column displays the word most likely to be predicted by humans, taking into account synonyms in our experiments. In contrast, the 'Inappropriate' column lists some incorrect word completions that fall within the same category as the expected word. The inappropriate completion is used to examine whether LMs will prioritize unsuitable completions that share a semantic category with the expected completions.

4.2 ROLE-88

It comprises 88 contexts, with one sentence per context designed to target role reversal. (Chow et al., 2016) illustrated the example in Table 3, "Completing the sentence requires semantic role identification and event knowledge, which means finding the accurate words associated with events and actions to fill in the blank". Although each completion (e.g., 'served') is suitable for only one of the noun orders and not the reverse, we use this diagnostic to test whether Arabic BERT models will face difficulty distinguishing appropriate continuations based on word order and semantic role.

4.3 NEG-SIMP-136

This diagnostic targets understanding the meaning of negation and category membership (Fishler et al., 1983). Table 4 presents a negation example along with its corresponding translation. The affirmative sentence allows us to assess the model's capacity to associate nouns with their hypernyms. Through this diagnostic, we investigate the model's ability to distinguish between affirmative and negative sentences, specifically whether it outputs the same word as in the affirmative case, as indicated in the 'match' column, or a different word, as shown in the 'mismatch' column.

4.4 NEG-NAT-136:

This diagnostic targets naturally occurring negative sentences and was derived from a human study conducted by Nieuwland and Kuperberg (2008). Building upon the experiment conducted by Fishler et al.,1983, it involves the creation of affirmative and negative sentences chosen to be more 'natural for somebody to say,' contrasting these with the non-natural affirmative and negative sentences. Table 5 shows an example of NEG-NAT-136 and its Arabic translation.

5 Experiments and Discussion

In this section, we analyze the results of running the diagnostics on AraBERTv2$_{base}$, AraBERTv2$_{large}$, and mBERT. We compare these results with those of the English BERT as presented in the paper "What BERT Is Not". We manually reviewed the results to ensure accuracy and to avoid instances where the language models provided correct answers not present in the diagnostic dataset.

5.1 Results for Common Sense and Pragmatic Inference

Figure 2 illustrates the performance of AraBERT$_{base}$, AraBERT$_{large}$, mBERT, BERT$_{base}$, and BERT$_{large}$ on the CPRAG-102 dataset, in terms of accuracy. It represents the percentage of items for which the 'expected' completion is among the model's top k predictions, with $k \in \{1, 5\}$. For accuracy at $k = 1$, both AraBERT$_{base}$ and mBERT achieved a score of 2.94%. In contrast, AraBERT$_{large}$ achieved more higher accuracy of 8.82% on the same task. On the other hand, BERT$_{base}$ and BERT$_{large}$ performed better with accuracies of 23.5% and 35.3%, respectively. This indicates that

[1] https://github.com/BasmaSayah/Psycholinguistic-Diagnostics-on-AraBERT

Table 2: Example of CPRAG-102 and its Arabic translation.

Context	Expected	Inappropriate
أرادت أن تجعل رموشها تبدو سوداء وسميكة حقًا. لذا طلبت من صديقتها أن تعيرها —————— She wanted to make her eyelashes look really black and thick. So she asked to borrow her older friend's ——	الماسكارا Maskara	أحمر الشفاه ـ قلادة lipstick | necklace

Table 3: Example of ROLE-88 and its Arabic translation.

Context	Completion
نسي صاحب المطعم أي زبون قام النادل ب —————— The restaurant owner forgot which customer the waitress had ———	خدمته served
نسي صاحب المطعم أي نادل قام الزبون ب —————— The restaurant owner forgot which waitress the customer had ———	خدمته served

Table 4: Example of NEG-SIMP-136 and its Arabic translation.

Context	Match	Mismatch
أبو الحنَاْء هو ——————	طائر	شجرة
A robin is a ———	bird	tree
أبو الحنَاْء ليس ——————	طائر	شجرة
A robin is not a ———	bird	tree

AraBERT$_{base}$ and mBERT do not perform well in common-sense and/or pragmatic inference tasks, while AraBERT$_{large}$ performs substantially better.

At $k = 5$, mBERT achieved the lowest accuracy of 5.88%, followed by AraBERT$_{base}$, which showed an improvement with an accuracy of 17.6%. AraBERT$_{large}$ achieved the highest accuracy in Arabic, reaching 23.52%. In the English part, both BERT$_{base}$ and BERT$_{large}$ achieved a 52.9% accuracy. The low accuracy scores highlight clear weaknesses in AraBERT's ability to handle common-sense and/or pragmatic inference.

Regarding completion sensitivity, Figure 2 illustrates the performance of AraBERT, mBERT, and BERT on the CPRAG-102 dataset in terms of sensitivity. This metric represents the percentage of items for which the model assigns a higher probability to the expected completion (e.g., 'Maskara,' as shown in Table2) than to any of the inappropriate completions (e.g., 'lipstick' or 'necklace'). mBERT assigns the highest probability to the expected completion only 5.88% of the time, whereas AraBERT$_{base}$ and AraBERT$_{large}$ achieve this 17.65% and 20.59% of the time, respectively. On the contrary, BERT$_{base}$ and BERT$_{large}$ exhibit a high sensitivity of 73.5% and 79.4% with English. This suggests that both versions of AraBERT and mBERT do not exhibit sensitivity in differentiating between good and bad completions within the same semantic category, with AraBERT noticeably better than the latter.

Upon introducing the threshold on the probability difference, mBERT's sensitivity remains the same, while AraBERT$_{base}$ and AraBERT$_{large}$ sensitivity drop slightly to 14.7% and 17.65%, respectively. In contrast, BERT$_{base}$ and BERT$_{large}$ sensitivity drop drastically to 44.1% and 58.8%. This still indicates that AraBERT lacks sensitivity in distinguishing between good and bad completions, whereas BERT$_{base}$ and BERT$_{large}$ exhibit some sensitivity, albeit with a small probability difference.

The qualitative analysis of the sentences where AraBERT$_{large}$ has made incorrect predictions shows that AraBERT$_{large}$ fails not only in one but in both common sense and pragmatic inference. In the phrase

أراد بابلو قطع الخشب الذي اشتراه لصنع بعض الرفوف. سأل جاره إذا كان
بإمكانه أن يعيره

meaning 'Pablo wanted to cut the lumber he had bought to make some shelves. He asked his neighbor if he could borrow her', AraBERT$_{large}$ predicted words related to wood but did not suggest 'saw.' This suggests that it recognized that the word to be predicted was related to the first sentence, succeeding in pragmatic inference, but failed to recognize what it was, indicating a failure in common sense understanding. In the phrase

إنها تستمر في تدويرها و تدويرها حول عنقها. يبدو أن ستيفاني سعيدة حقًا لأن
دان أعطاها ذلك ——————

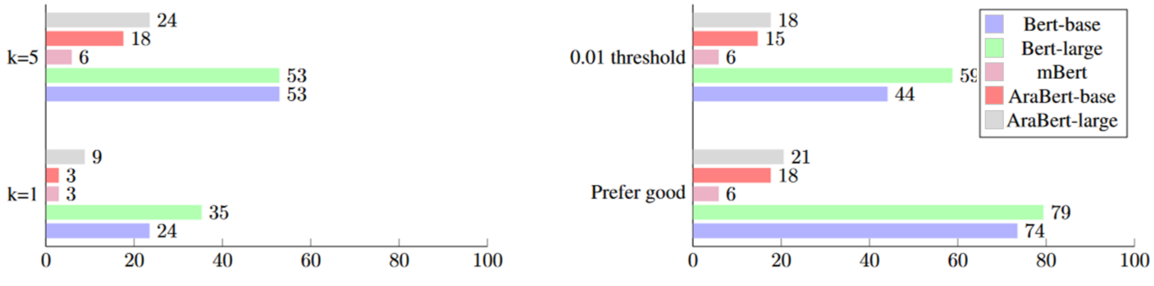
meaning "She keeps twirling it around and around under her collar. Stephanie seems really happy that Dan gave her that ——', AraBERT$_{large}$ predicts the words 'place' and 'time.' This indicates that it only used the second sentence for predictions, failing in pragmatic inference.

5.2 Results for role reversals and event knowledge

As demonstrated in Figure 3 when $k = 1$, mBERT exhibits poor performance (a 0% accuracy). This

Table 5: Example of NEG-NAT and its Arabic translation.

Context	target_aff	target_neg
مع المعدات المناسبة، يعد الغوص تحت الماء ———	آمن	خطير
With proper equipment, scuba-diving is very ———	safe	dangerous
مع المعدات المناسبة، لا يعد الغوص تحت الماء ———	آمن	خطير
With proper equipment, scuba-diving isn't very ———	safe	dangerous



(a) Accuracy.

(b) Sensitivity.

Figure 2: Performances of BERT, mBERT and AraBERT on the CPRAG-102 dataset.

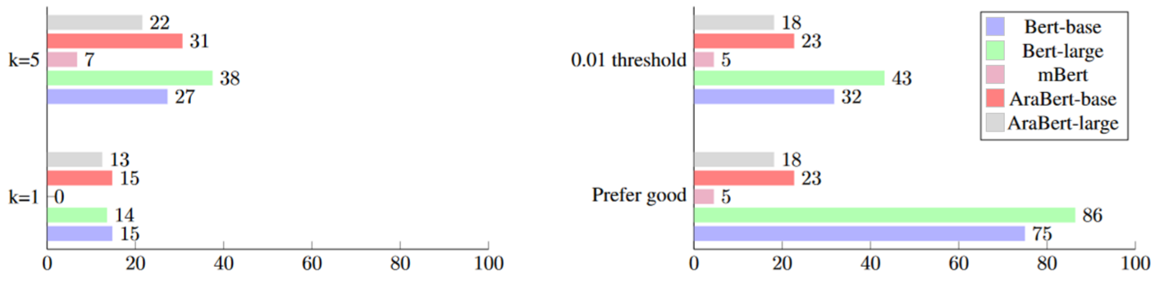
suggests that mBERT is not suitable for role reversal and event knowledge tasks. In contrast, BERT$_{base}$ and AraBERT$_{base}$ show similar accuracies, both approximately at 14.8%. However, AraBERT$_{large}$ and BERT$_{large}$, despite their larger architectures, achieve slightly lower accuracies of 13.6% and 12.5%, respectively. These results indicate that AraBERT$_{large}$ and BERT$_{large}$ may benefit from further fine-tuning tailored to these tasks, emphasizing that model size alone does not guarantee enhanced performance.

When $k = 5$, mBERT still lags with an accuracy of 6.81%. AraBERT$_{base}$ and AraBERT$_{large}$ both demonstrate improved accuracies, with AraBERT$_{base}$ surprisingly surpassing AraBERT$_{large}$. AraBERT$_{base}$ achieves an accuracy of 30.68%, while AraBERT$_{large}$ achieves 21.59%. Although AraBERT$_{base}$ and AraBERT$_{large}$ exhibit better performance than mBERT for $k = 5$, they are still outperformed by the English-language models BERT$_{base}$ and BERT$_{large}$, which achieved accuracies of 27.3% and 37.5%, respectively. Considering a larger number of predictions enhances accuracy for all models. However, English-language models BERT$_{base}$ and BERT$_{large}$ consistently outperform the multilingual and Arabic-specific models in this task.

Figure 3 illustrates the sensitivity of BERT models to role reversals. mBERT performs poorly for Arabic, exhibiting a sensitivity of only 4.54%. AraBERT$_{base}$ and AraBERT$_{large}$ show moderate sensitivity, with AraBERT$_{base}$ at 22.72% and AraBERT$_{large}$ at 18.18%. In contrast, BERT$_{base}$ and BERT$_{large}$ demonstrate high sensitivity to "good completions" with accuracies of 75% and 86.4%, respectively.

After introducing the threshold of 0.01, mBERT maintains a low sensitivity of 4.54%. AraBERT$_{base}$ and AraBERT$_{large}$ also maintain their sensitivities at 22.72% and 18.18%, respectively, while BERT$_{base}$ and BERT$_{large}$ maintain relatively higher sensitivities at 31.8% and 43.2%, respectively. Overall, the results suggest that mBERT for the Arabic language is not well-suited for role reversals and/or event knowledge tasks. The moderate sensitivity of AraBERT models indicates their ability to identify "good completions" to some extent. In contrast, the English-language models, particularly BERT$_{large}$, exhibit better performance in these tasks, highlighting potential challenges in adapting these models for Arabic language tasks or the need for further fine-tuning.

During manual analysis of sentences where mBERT, AraBERT$_{base}$ and AraBERT$_{large}$ failed, all models frequently produced the unknown token, indicating challenges in generating predictions for the given contexts. In cases where words were generated, mBERT's predictions often lacked coherence and didn't make sense whereas AraBERT$_{base}$ and AraBERT$_{large}$ produced logically consistent predictions that differed from those generated for their role-reversed versions of the sentence. This suggests that mBERT struggles with producing meaningful predictions. Conversely, the limitations



(a) Accuracy.

(b) Sensitivity.

Figure 3: Performances of BERT, mBERT and AraBERT on the ROLE-88 dataset.

of the AraBERT models appear to be primarily related to event knowledge, as they generate words that are logically consistent within the sentence context but struggle to predict the accurate word related to the event or action. Furthermore, AraBERT$_{large}$ showed more uncertainty than AraBERT$_{base}$, indicating AraBERT$_{large}$ need for additional context. However, its responses were grammatically more accurate compared to AraBERT$_{base}$.

5.3 Results for negation understanding

Figure 4 illustrates the accuracy of BERT models in predicting affirmative and negative sentences. Affirmative sentences were used to evaluate BERT's ability to associate nouns with their hypernyms(categories). When we examine the accuracy scores for affirmative sentences, we see mBERT achieved the lowest accuracy, scoring 0%, indicating it is not suitable for category membership prediction. Both AraBERT$_{base}$ and AraBERT$_{large}$ achieved accuracies of 44.44% and 33.33% respectively, in predicting category membership. While English-language models BERT$_{base}$ and BERT$_{large}$ achieved a perfect accuracy of 100%. This suggests that AraBERT models are less effective in category membership prediction compared to the highly effective English models.

In the case of negative sentences, mBERT achieved an accuracy of 0%, which is evident given that it also failed with affirmative sentences. AraBERT$_{large}$ also achieved an accuracy of 0% in understanding negation, similar to BERT$_{base}$ and BERT$_{large}$. This result suggests these models' failure to understand negations. On the other hand, AraBERT$_{base}$ achieved a relatively low accuracy of 5.55% in understanding negation. The correct results it obtained may be attributed to its potential understanding of negation or pattern recognition.

After checking predictions manually, AraBERT mostly gives the same predictions for positive and negative sentences, except for one sentence which is

السلمون المرقط من مجموعة ———

meaning 'A trout is———' AraBERT$_{base}$ predicted "fish" as a first prediction for the affirmative statement but provided a different answer, "chicken," for the negative statement. AraBERT$_{large}$, on the other hand, did not exhibit this distinction.

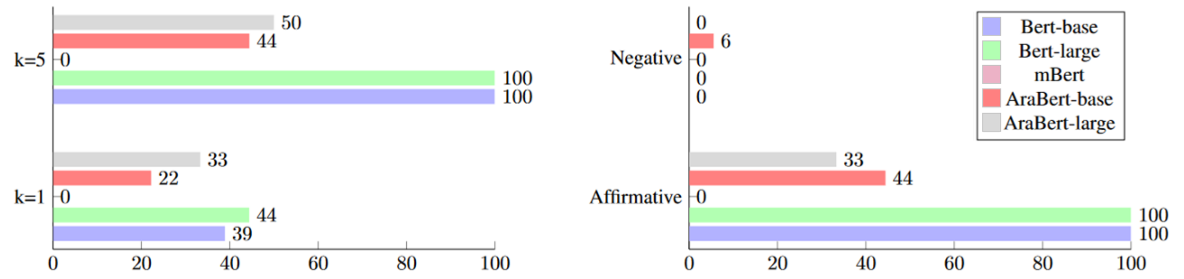
Figure 5 Illustrates the accuracies of BERT models for natural affirmative and negative sentences, with the distinction that these affirmative sentences do not test category membership. Regarding affirmative sentences, AraBERT$_{large}$ emerges as the top-performing model in this context, achieving an accuracy of 87.5%, closely followed by BERT$_{large}$ with an accuracy of 75%. BERT$_{base}$ and AraBERT$_{base}$ achieved moderate accuracies of 62.5% and 68.75%, respectively. In contrast, mBERT failed to make any correct prediction, yielding an accuracy of 0%, indicating its instability in making predictions.

Turning to negative sentences, AraBERT$_{base}$ and AraBERT$_{large}$ showed moderate performance, achieving accuracies of 43.75% and 50%, respectively, while BERT$_{base}$ and BERT$_{large}$ demonstrated strong performance with accuracies of 87.5% and 100%.

When examining the top Predictions of AraBERT$_{large}$, they all align with each other and do not contradict each other, whether for affirmative or their corresponding negative sentences, This consistency suggests that there is an opportunity to improve how the model handles negation.

6 Conclusion

In this study, we examined the capabilities of multilingual BERT for Arabic, as well as AraBERT base and large versions, using psycholinguistics. While AraBERT is better than Multilingual BERT, it has



(a) Accuracy.

(b) Sensitivity.

Figure 4: Performances of BERT, mBERT, and AraBERT on the NEG-SIMP-136 dataset.

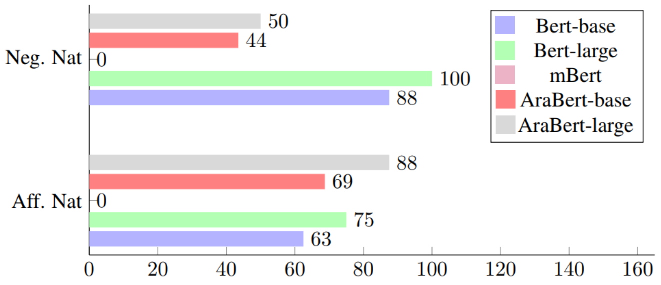


Figure 5: Performances of BERT, mBERT, and AraBERT on the NEG-NAT-136 dataset.

notable weaknesses in common sense and pragmatic inference. In this task, the large version consistently outperforms the base version of AraBERT. Additionally, AraBERT faces challenges in recognizing words related to events and actions, where the base version consistently outperforms the large version. In negation tasks, both AraBERT models often struggle to distinguish affirmative from negative sentences, except in rare cases, marking an improvement compared to English BERT models that do not make this distinction at all. All models perform well with natural negative sentences, likely relying on pattern recognition rather than a deep understanding of negation cues. This situation presents opportunities for enhancing language models' grasp of negation. Further research is needed to fully understand each model's strengths and weaknesses, facilitating more informed decisions when choosing a language model.

References

Ahmed Abdelali, Fahim Dalvi, Hassan Sajjad, and Nadir Durrani. 2022. Post-hoc analysis of Arabic transformer models. In *Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 91–103, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Eman Albilali, Nora Altwairesh, and Manar Hosny. 2021. What does bert learn from arabic machine reading comprehension datasets? In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 32–41, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Hasna Chouikhi, Hamza Chniter, and Fethi Jarray. 2021. Arabic Sentiment Analysis Using BERT Model. In *Advances in Computational Collective Intelligence*, pages 621–632. Springer International Publishing.

Wing Yee Chow, Ellen Lau, Colin Phillips, and Cybelle Smith. 2016. A 'bag-ofarguments' mechanism for initial verb predictions. *Language, Cognition and Neuroscience*, 31(5):577–596.

Simone Conia and Roberto Navigli. 2022. Probing for predicate argument structures in pretrained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4622–4632, Dublin, Ireland. Association for Computational Linguistics.

Jacob Devlin, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Ming-Wei Chang. 2019. Bert: Pretraining 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.

Aakanksha Chowdhery et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Rishi Bommasani et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Kara Federmeier and Marta Kutas. 1999. A rose by any other name: Long-term memory structure and sentence processing. *Journal of Memory and Language*, 41(4):469–495.

Ira Fishler, Donald Childers, Salim Roucos, Nathan Perry, and Paul Bloom. 1983. Brain potentials related to stages of sentence verification. *Psychophysiology*, 20(4):400–409.

Trevor Harley. 2013. *The psychology of language: From data to theory*. Psychology press.

Zhengbao Jiang, Haibo Ding, Graham Neubig, Zhengbao Jiang, and Jun Araki. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Marta Kutas and Steven Hillyard. 1984. Brain potentials during reading reflect word expectancy and semantic association. *Nature*, 307(5947):161–163.

Bai Li, Guillaume Thomas, Yang Xu, Frank Rudzicz, and Zining Zhu. 2021. How is BERT surprised? layerwise detection of linguistic anomalies. 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 4215–28. Association for Computational Linguistics.

Vladislav Lialin, Kevin Zhao, Namrata Shivagunde, and Anna Rumshisky. 2022. Life after BERT: What do other muppets understand about language? In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3180–3193, Dublin, Ireland. Association for Computational Linguistics.

Timothee Mickus, Mathieu Constant, Kees van Deemter, and Denis Paperno. 2020. What do you mean, bert? assessing bert as a distributional semantics model. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 279–290, New York, New York. Association for Computational Linguistics.

Xuan-Bac Nguyen, Chi Nhan Duong, Xin Li, Susan Gauch1, Han-Seok Seo, and Khoa Luu. 2023. Micron-bert: Bert-based facial micro-expression recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1482–1492.

Mante Nieuwland and Gina Kuperberg. 2008. When the truth is not too hard to handle: An event-related potential study on the pragmatics of negation. *Psychological Science*, 19(12):1213–1218.

Matthew Peters, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer, and Mark Neumann. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the NAACL, Vol. 1*, pages 2227–2237, New Orleans, Louisiana. ACL.

Alec Radford, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pretraining.

Samuel Rönnqvist, Jenna Kanerva, Tapio Salakoski, and Filip Ginter. Is multilingual BERT fluent in language generation?

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. page 6000–6010.

Alex Wang, Julian Michael, Felix Hill, Omer Levy, Samuel Bowman, and Amanpreet Singh. 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.

An NLP Analysis of ChatGPT's Personality Simulation Capabilities and Implications for Human-centric Explainable AI Interfaces

Thorsten Zylowski and **Matthias Wölfel**
University of Hohenheim, Schloss Hohenheim 1, 70599 Stuttgart, Germany
Karlsruhe University of Applied Sciences, Moltkestraße 30, 76133 Karlsruhe, Germany
thorsten.zylowski@uni-hohenheim.de

Abstract

This paper explores the potential of ChatGPT in simulating diverse personalities for application in adaptive human-centric eXplainable Artificial Intelligence (XAI) interfaces. A dataset of 4329 text datapoints across 13 simulated personalities from ChatGPT were collected. Extensive linguistic analyses were conducted using metrics from Natural Language Processing (NLP) for basic linguistic features, readability, lexical richness, and sentiment. Additionally, a personality classifier was trained with a F1-score of 0.79 to understand which personalities are unique in wording and style. This was further substantiated through the application of the SHAP (SHapley Additive exPlanations) framework, which unveiled important words in personality classification. It was found that ChatGPT is capable of simulating several levels of professionalism as well as more emotional personalities that adapt human characteristics and can be used in human-centric XAI interfaces, although specific user testing is still pending.

1 Introduction

With ChatGPT, large language models (LLM) and generative artificial intelligence (generative AI) are entering daily life at a speed never before seen with any other technology. ChatGPT contributes to empowering people and assists with many everyday and professional tasks. One reason for this high adoption rate is the chosen interface in form of a conversation, which is a natural way of interaction. When a question is asked, ChatGPT tries to provide accompanying descriptions and explanations. However, it is also known that ChatGPT tends to provide false information and express it confidently and also make up facts and sources. This effect is referred to as hallucination. (Ji et al., 2023) ChatGPT is known for generating high-quality texts for various situations, but it is also known to differ in its wording from people. (Mitrović et al., 2023)

Due to its ability to produce well readable and high quality text, ChatGPT has great potential to assist in the development of human-centered explainable artificial intelligence (XAI) interfaces, which help to increase trust in artificial intelligence (AI) systems by making AI decisions more transparent. Many machine learning (ML) models still have the problem that they are opaque and cannot be explained. The reason for this is the black-box nature of these models, which makes it impossible for a human to understand the decision paths of the model. One task of XAI is to extract information from the model that can be provided to a human so that the model's decisions can be understood. In addition, it is of high importance *how* the explanation is given.

Miller (2019) describes that a good explanation is *social*, referring to work by Hilton (1990), according to whom a *"causal explanation is first and foremost a form of social interaction."* Consequently, it is important for a good explanation *who* the explainer is, who the receiver of the explanation is (explainee) and what the context is. For example, it makes a difference whether a professor is explaining something to another professor within the same research field, or whether he/she is explaining something to a student. The way the explanation is given differs in both situations. In contrast, ChatGPT responds to every request in the same way, unless prompted otherwise. It has no information about who it is having the conversation with unless it is made aware of it. Miller (2019) further elaborates on the work of Hilton (1990), who describes that a causal explanation is always a *conversation*. Accordingly, it would be desirable to have XAI interfaces capable of generating explanations in natural language and adapted to the situation and to the human in form of a conversation. This is further reinforced by the fact that people demand that a good explanation can adapt to their needs. (Zylowski, 2022) This includes, among other things, the ability to get explanations on demand and in different formats and granularities. Before the developments in the field of LLM, developing a conversational human-centric XAI interface that can be adjusted to the requirements of a person was very difficult or impossible and the impact of intent-based conversational interfaces were limited. (Jentzsch et al., 2019) Even if it was known what a good explanation to a person should look like, it was technically very difficult to actually create an adequate explanation. With the potential of ChatGPT to simulate different personalities, it is possible to develop XAI interfaces that can be adapted to different people and to different needs and requirements of those people. However, it is still an open question how well ChatGPT can simulate different personalities and how well responses are adapted to people's needs.

This paper investigates the ability of ChatGPT to simulate different personalities and describes the advantages for adaptive human-centric XAI interfaces. An NLP approach is chosen by applying different metrics to ChatGPT texts in different personality styles and it is investigated how clearly these personalities can be distinguished from each other and which phrases and words are typical for different personalities. By exploring the potential of adapting explanations to individuals, this work aims to address the current limitations and unlock the full potential of ChatGPT in fostering trust and transparency in AI systems.

2 Related Work

ChatGPT's responses are currently being studied by many researchers and the applicability in different domains is being validated. It is investigated whether texts generated by humans can be distinguished from those generated by ChatGPT and what the differences are. Mitrović et al. (2023) investigate whether a classifier can be trained to distinguish human-generated texts from those generated by ChatGPT, achieving 79% accuracy. Through an analysis of the classifier with the XAI framework SHAP, they look for differences between the formulations. They find that ChatGPT tends to focus on describing experiences rather than expressing feelings and it avoids using personal pronouns. Moreover, it has a tendency to utilize uncommon or unusual words and never employs aggressive language or rude vocabulary in its responses. Mindner et al. (2023) created several text classifiers for the educational field to distinguish texts generated and rephrased by ChatGPT from human-created texts, with F1-scores of over 96% and 78%, respectively, outperforming even GPTZero[1], the most prominemt approach, in the best *basic* text rephrasing task. Other authors focus on how trustworthy the texts generated by ChatGPT appear to people. Li et al. (2023) analyze the applicability of ChatGPT for Information Extraction (IE) tasks and found that ChatGPT performs poorly on the Standard-IE setting, but performs very well on the Open-IE setting. Furthermore, they investigated the quality and trustworthiness of the explainations of ChatGPT responses in a self-check and by domain experts and judged them to be of high quality and trustworthy.

One aspect that is not yet investigated in current studies is the adaptability of the formulations of ChatGPT in different situations and under different user requirements. For effective use in human-centered XAI interfaces, ChatGPT must be able to generate different types of formulations that are adapted to people's needs. It is important that interfaces also address humans on an emotional level to enable trust. The fact that ChatGPT tends not to express emotions (Mitrović et al., 2023), can be challenging in this regard.

3 Method

This section presents the approach including data collection and metrics used.

3.1 Data Collection

For this study, a total of 333 instructions were manually selected from the ShareGPT[2] dataset which contains real world examples of conversations with ChatGPT. For the selection process a set of 500 randomly selected instructions was created. The instructions were then manually filtered based on usefulness (i.e. instructions that were not written in English or that consisted of only one word or that contained only a technical command were removed). The instructions were then utilized to interact with the ChatGPT API. The *gpt3.5-turbo* model was selected for the data collection process, because it was the best accessible model at the time. To ensure a comprehensive analysis, ChatGPT was requested to respond to the instructions, in addition to the default answer, using 10 distinct personality

[1] https://gptzero.me/

[2] https://sharegpt.com/

styles as described in Section 3.2. In order to cater to different user groups commonly encountered in XAI interfaces, two additional styles were incorporated — one targeting laypeople and the other aimed at experts. Thus, a total of 13 different styles were applied during the interactions and resulted in the acquisition of 4329 datapoints.

3.2 Personality Styles

In order to address people in different ways in an XAI interface, besides ChatGPT's default style, 10 personality styles were created to satisfy individual needs and requirements. Additionally the two target groups *laypeople* and *expert* are described. When selecting the personality styles, attention was paid to a diverse range and existing findings were taken into account (e.g. addressing laypersons and experts). The styles should contain human characteristics that can be useful for an explanation. However, the real usefulness still has to be determined in user experiments.

Default: ChatGPT's default personality if no specific prompt to change its personality is given.

Child-like: Simple, short and playful and targets a younger audience or can be helpful when explaining basic concepts to users who prefer a more lighthearted and approachable explanation.

Parent-like: A parent-like explanations could provide patience and empathy and may offer guidance and support throughout the learning process.

Professorial: High level of expertise and use academic language. Those styles could be useful when catering to users who appreciate in-depth knowledge and a more formal style of explanation.

Friendly Companion: Supportive, uses conversational tone, engages in conversations and listens actively to user queries.

Expert Guide: Talks in a knowledgeable and authoritative manner and can be effective when users are seeking accurate and detailed information from a trusted source.

Storyteller: Focus on narratives and anecdotes and can give a more memorable experience, enabling users to connect with the AI through storytelling.

Helpful Assistant: Creates clear and concise explanations and emphasizes practicality and utility.

Humorous & Entertaining: Uses jokes, puns, or witty remarks and can make the interaction more enjoyable and help alleviate potential boredom or monotony during the explanation process.

Motivator: Inspiring and encouraging users and can provide positive reinforcement, acknowledge progress, and instill confidence in users' ability to grasp the material.

Technician: Pays attention to detail and on the technical aspects and can be valuable for technical professionals.

Laypeople: Aims at laypersons and will try to present content in a way that is easy to understand.

Expert: Targets experts and will provide a lot of expert knowledge.

The personality styles can be roughly divided into two categories. One category includes more technical and professional personality styles (ChatGPT's default, professorial, expert, expert guide, technician). The other category includes emotional, human-oriented personality styles (helpful assistant, storyteller, motivator, laypeople, friendly companion, humorous/entertaining, parent-like, child-like). The two categories are not strictly separated and overlap of personality styles is possible.

3.3 Prompts

A separate prompt was created for each personality style, with most of them following the scheme:

"I want you to communicate like a <personality> in the following conversation. I will give you a question or instruction and I want you to answer it in a way a <personality> would do it.

<instruction>"

For personality styles like child-like or professorial the personality tag was replaced with words like *"child"* or *"professor"*. In cases where this was not possible, the prompt was adjusted to instruct ChatGPT to act in a specific way, e.g. *"I want you to communicate in a humorous/entertaining way [...]"*. The two prompts, aimed at laypeople and experts, follow the scheme:

"I want you to communicate in such a way that your answers are directed at laypeople/experts. I will give you a question or instruction and I want you to answer it in a way that laypeople/experts can understand.

<instruction>"

The instruction tag was replaced by an instruction from ShareGPT.

4 Metrics

Metrics from the fields of NLP and linguistics were used to analyze the responses of ChatGPT. In addition, a ML model was trained that attempts to classify the different personality styles based on the texts. An XAI framework was then used to examine specific words and phrases of these styles.

4.1 Linguistic Metrics

For the linguistic analysis of ChatGPT's responses the spaCy framework[3] in version 3.6 was used. The *average token-wise text length* was calculated to investigate if there are differences in the length of the answers between the personalities. To check how direct and precise a text is written, the *average number of stopwords* was determined. Personalities expected to produce text that is low in information density are likely to have an increased number of stopwords. These could include, for example, the storyteller and the motivator personalities.

Further differentiation of responses could be possible by the *average number of named entities* used. The more precise an answer is, the more named entities it might contain. A more professional answer will most likely contain more facts and details that could also be expressed in named entities. To compensate for the dependence on the length of the texts, the average number of stopwords and named entities per 100 words was calculated.

4.2 Readability

The readability of the responses in the different personalities of ChatGPT is expected to differ significantly. For this reason, the readability was compared using the well established Flesch Reading Ease (FRE) index (Flesch, 1948). The index has a range from 0 to 100, where a value of 0 represents very hard to read text and 100 represents very easy to read text. The FRE is derived from a base value from which the weighted average sentence length and the weighted average number of syllables in a word are subtracted.

4.3 Lexical Richness

Lexical richness is classically formed as the token-type ratio (TTR), which is the relation of unique words (types) and the set of total words (tokens) (Templin, 1957). There are several variants of this measure that make corrections to compensate for a dependence on the length of the text (Torruella and Capsada, 2013). One of these measures is the Measure Of Textual Lexical Diversity (MTLD) as described in McCarthy and Jarvis (2010). For the calculation, the text is divided into segments for which the TTR is calculated. A segment is expanded until a TTR of a given threshold is reached. Then the number of words in the text is divided by the number of segments to calculate the lexical richness. A higher MTLD suggests that the text uses a wider variety of words across its segments, and therefore has greater lexical richness. A lower MTLD indicates that the text may have more repetition and less varied vocabulary.

4.4 Sentiment Analysis

Personality styles that formulate text on an emotional level (e.g. motivator personality) can be expected to show increased positive or negative sentiment. In order to investigate whether personalities exhibit a certain sentiment, a sentiment analysis of the texts was performed. The model used is *distilbert-base-uncased-finetuned-sst-2-english* which is a DistilBERT model (Sanh et al., 2020) fine-tuned on SST-2 dataset (Socher et al., 2013). The classification results in an assignment of a label POSITIVE or NEGATIVE to the text with a percentage indication of the strength of the sentiment. To distinguish positive and negative sentiments numerically, all negative sentiments were converted to a negative value. Thus, all positive sentiments run from 0 to 1 and all negative sentiments from 0 to -1.

4.5 Personality Classifier

To gain a deeper understanding of the ChatGPT texts, a classifier was trained that attempts to predict the respective personality based on the texts.

[3] https://spacy.io/

It is reasonable to assume that there are personalities that are very easy to predict because they have unique phrases and style. Other personalities are more similar to each other and more difficult to predict. For the training, a *distilbert-base-uncased* model was fine-tuned on the 4329 data points with 20% test data, 5 epochs of training, a learning rate of 0.00002 and a batch size of 16.

4.6 Explaining the Classifier

The personality classifier was examined using the XAI framework Shapley Additive Explanations (SHAP) (Lundberg and Lee, 2017) to analyze which words are typical of the different personalities. SHAP is based on the shapley values of cooperative game theory and attempts to assign a value to each feature, in the case of text each token, indicating how much contribution that feature has to the overall classification.

5 Results

Presented below are the results of the NLP analysis of the 4329 data points split between the 10 defined personalities, the two specific target groups of laypeople and experts, and the default response of ChatGPT. For simplified readability, the following will always refer to 13 personalities.

5.1 Text Length

The average text length of the responses for the different personality styles is shown in Figure 1. The length of ChatGPT's default response is in the upper range of values with a average text length of 332 tokens. The child-like personality has the shortest average length with 148 tokens, while the storyteller personality has the longest with 468 tokens on average. The more professional/technical styles are in the upper range of values.

5.2 Stopwords

Figure 2 shows the average number of stopwords for each personality style with ChatGPT's default answer at the second position with 31 stopwords per 100 words. It can be seen that the technical/professional personality styles are in the lower range of stop words and the more personal/emotional styles are in the upper range with child-like (42 stopwords per 100 words) and parent-like (43 stopwords per 100 words) at the top.

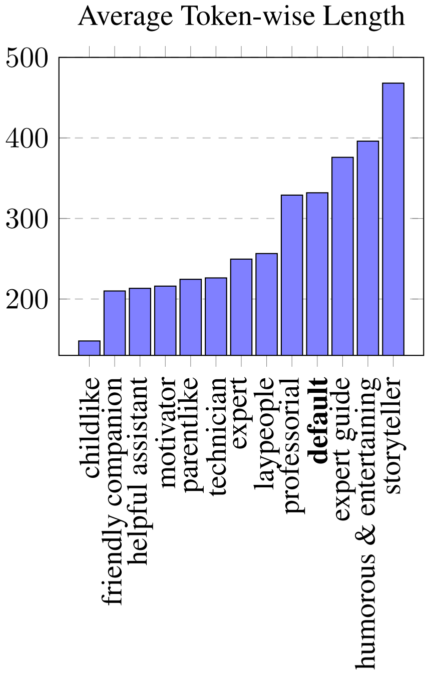


Figure 1: The average token-wise lengths of the personality styles.

5.3 Named Entities

The average number of named entities is shown in Figure 3. The lowest number of named entities occur for the personality styles motivator with 2.22 named entities per 100 words and storyteller with 2.23 named entities per 100 words, the highest for ChatGPT's default response with 4.42 named entities per 100 words. The technical styles also tend to be on the higher end for the number of named entities. Surprisingly, the professorial style is an exception.

When the named entities are split according to their categories, it can be seen that the motivator personality uses almost no numbers and the default personality of ChatGPT uses no ordinal entities, such as *"first"*, *"second"*, *"third"*, etc. The entertaining personality contains the most named entities with the category WORK OF ART, which classifies book tiles, song names etc. The expert personality has a high value for the FAC category which contains building, airports, highways etc.

5.4 Readability

The scores for the Flesch Reading Ease index for the different personality styles are shown in Figure 4. Low values mean that a text is more difficult to read. The technical and more professional styles, including ChatGPT's default response, are on the

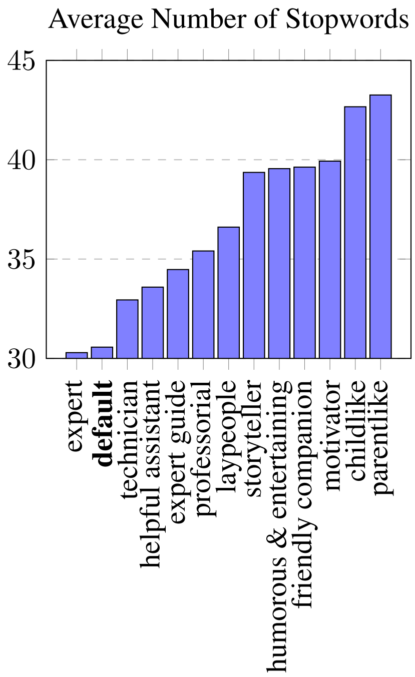


Figure 2: Average number of stopwords of the personality styles per 100 words.

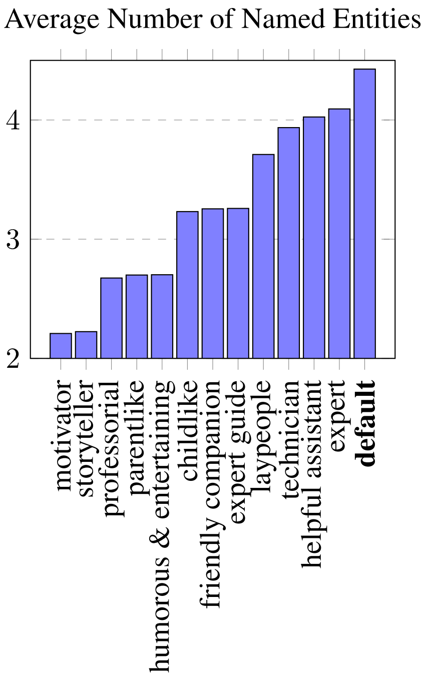


Figure 3: Average number of named entities of the personality styles per 100 words.

lower end. The professorial style is the hardest to read with a score of 19.12, while the child-like is the easiest with a score of 77.99. For all personalities, the average is below 78, which means that on average none of the texts are easy or very easy to read.

5.5 Lexical Richness

The distribution of the MTLD lexical richness scores is shown in Figure 5. ChatGPT's default answer has the lowest MTLD score (71.0), similar to the child-like personality, which means, that the lexical richness is low. The highest value is for the humorous/entertaining personality (129.0). No distinction can be made between technical/professional and non-technical/non-professional as in the other results. Lexical Richness seems to be a very individual property of the respective personality styles.

5.6 Sentiment Analysis

In Figure 6 is shown, that the average sentiment scores ranges from 0 to 1, with ChatGPT's default personality having a value close to 0. This does not mean that this personality generates very neutral texts. The opposite is true, as can be seen in Figure 7. The distribution of negative and positive sentiments balance each other, resulting in a neutral value on average. In fact, all the personalities behave in this way, with differing weights. For example, the motivator personality has the predominant amount of sentiments in the upper positive range. In particular, there is no negative trend in sentiment. All changes in the weights are in the direction of more positive sentiment. Texts that contain code tend to be classified with a negative label.

5.7 Evaluating the Personality Classifier

The personality classifier to decide a personality style out of the 13 classes reached a F1-score of 0.79. Although the value is already quite good, a difference can be seen between the individual personality styles. As suspected, there are styles that are particularly predictable. Humorous/entertaining, storyteller and parent-like personalities with a F1-score of 0.97, child-like with 0.95 and professorial with 0.89. These personality styles have very unique formulations and style. The storyteller personality in particular has frequent unique phrases, such as *"once upon a time"*, that are not used by other styles. The hardest to predict personalities are technician with F1-score of 0.48, expert with 0.61 and laypeople with 0.63.

A look at the confusion matrix, which can be seen in Figure 8, shows that the technician personality is in 15 cases confused with the expert personality and 6 times with the helpful assistant.

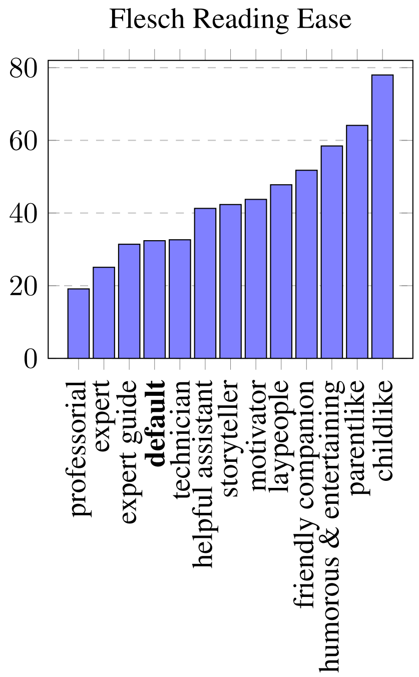


Figure 4: Flesch's Reading Ease (FRE) values of the personality styles. FRE ranges from 0 to 100, with higher values indicating better readability.

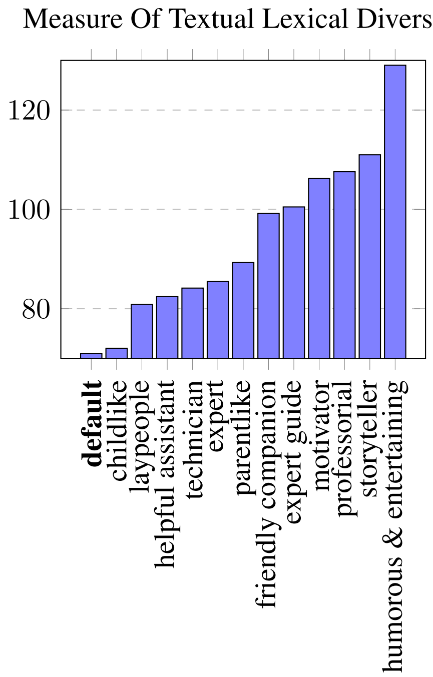


Figure 5: Measure Of Textual Lexical Diversity (MTLD) values of the personality styles, with higher values indicating higher lexical richness.

The expert personality is confused in 8 cases with the technician, in 7 cases with the helpful assistant and in 5 cases with ChatGPT's default answer. The laypeople personality is in 8 cases confused with the helpful assistant, in 6 cases with ChatGPT's default answer and in 2 cases with the technician. This shows very clearly that these personality styles are very similar to each other and may use the same phrases.

5.8 Extracting Important Words with SHAP

The most important words for the prediction of personalities with the personality classifier were extracted with the SHAP XAI framework. A global explanation approach was chosen using summarized text explanations. The extracted top 10 words for each personality style are shown in Table 1.

6 Discussion

The lengths of personality styles generated by ChatGPT are as expected. A child-like personality is expected to have rather short texts, since many details are omitted. In contrast, the storyteller personality generates very long texts because whole stories are formulated with embellishments. An explanation for why the professional/technical personality styles tend to generate longer texts is that they contain more factual content, are described in more detail, and contain code. The average number of stopwords, the average number of named entities, Flesch's Readining Ease and the sentiment score can be explained by the categorization into professional/technical and non-professional/non-technical personality styles. Personality styles that are not so technical/professional but more human-oriented, take into account other dimensions besides the factual content, e.g. include motivational phrases, descriptive texts, entertaining passages, etc. This leads to the increased number of stopwords. At the same time, due to a different focus (motivation, entertainment, etc.) the facts are reduced, which explains the reduced number of named entities. As professionalism and technical focus increase, wording becomes more complex, which degrades readability, explains the falling Flesch Reading Ease, and aligns with expectation. ChatGPT's ability of being able to switch appropriately between professional/technical and emotional/human-centered formulation of texts through the presented prompts fits very well with the requirement for adaptive human-centered XAI interfaces to adapt to user needs and to provide information in different preparations and different granularities.

The distribution of MTLD indicates that the lexical richness of personality styles has no simple ex-

Personality	Most Important Tokens
Default	“Certainly”, “steps”, “bu”, “B”, “template”, “:”, “Firstly”, “you”, “Python”
Child-like	“!”, “Oh”, “.”, “and”, “,”, “”, “friend”, “Hi”, “or”
Parent-like	“Parent”, “.”, “?”, “Child”, “,”, “sweetie”, “a”, “!”, “sweetheart”
Professorial	“Indeed”, “pleased”, “scholarly”, “.”, “,”, “Thank”, “Colleagues”, “intriguing”, “excellent”
Friendly Companion	“absolutely”, “delighted”, “Hey”, “,”, “fascinating”, “Oh”, “course”, “!”, “assist”
Expert Guide	“walk”, “guide”, “Welcome”, “welcome”, “Certainly”, “Allow”, “.”, “!”, “explorer”
Storyteller	“Once”, “nestled”, “!”, “expertise”, “time”, “tale”, “upon”, “a”, “qui”
Helpful Assistant	“course”, “assist”, “helpful”, “Certainly”, “assistant”, “,”, “.”, “!”, “As”
Humorous/Entertaining	“!”, “Oh”, “jolly”, “Well”, “?”, “,”, “comrades”, “.”, “Don”
Motivator	“!”, “welcome”, “Absolutely”, “Remember”, “inspire”, “amazing”, “you”, “,”, “friend”
Technician	“technician”, “Technician”, “assist”, “,”, “Sure”, “.”, “and”, “V”, “X”
Laypeople	“Certainly”, “Sure”, “”, “Singapore”, “Remember”, “,”, “guide”, “memory”, “appropriate”
Experts	“Certainly”, “examples”, “Experts”, “expert”, “,”, “experts”, “subjective”, “I”, “Title”

Table 1: Most importants words for each personality extracted from the personality classifier using XAI framework SHAP.

planation and that a separate explanation for each style needs to be found in future work. However, for the personality styles motivator, professorial, storyteller, and humorous/entertaining, which are in the upper range of values, the result is at least plausible, since many unique words for these can be expected. If the scores of the MTLD are compared with the Flesch Reading Ease, there are cases, such as the professorial personality, where the texts are very difficult to read and have a high lexical richness. A random examination of the data shows that the texts are indeed particularly written in a sophisticated way. There are other cases, such as ChatGPT's default personality, which is also difficult to read, but has a low lexical richness. Possible explanations could be a more complex sentence structures, advanced vocabulary and redundancies.

Sentiment analysis showed that for personality styles for which positive sentiments are expected (motivator, storyteller, friendly compation, humorous/entertaining) the texts were adequately simulated by ChatGPT. This is due to an increased use of words with positive sentiment. The ability of ChatGPT to provide an explanation to a human in an appropriate sentiment is a strong feature for human-centric XAI interfaces. Explanations conveyed with an appropriate sentiment seem more natural and it can be presumed that trust is increased. The analysis of the personality classifier shows that there are personality styles that are very distinct from each other. The reason are words and phrases typical for the personality. This finding becomes even clearer by analyzing the words extracted with SHAP. For example, the professorial style uses sophisticated words such as *“indeed”*, *“pleased”*, *“intriguing”*, and *“excellent”*, which are very appropriate for this style. The friendly companion uses very friendly words and the expert guide personality is very welcoming. It can also be seen that some styles reference themselves. For example, the helpful assistant uses the words *“assist”* and *“assistant”*, the technician uses the word *“technician”*, and the expert style uses the words *“expert”* and *“experts”* frequently. This is because ChatGPT generates phrases like *“Ok I will give the answer like a technician”* at the beginning of the answer.

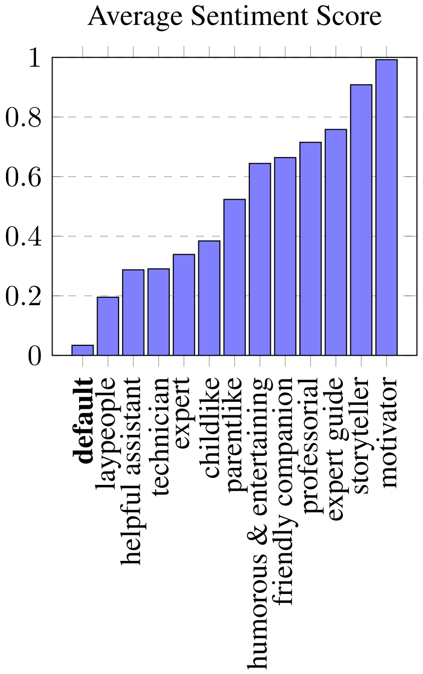


Figure 6: Average sentiment score for each personality style.

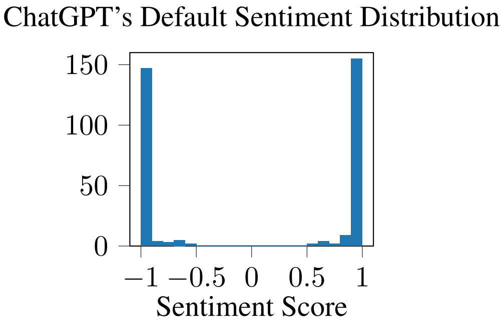
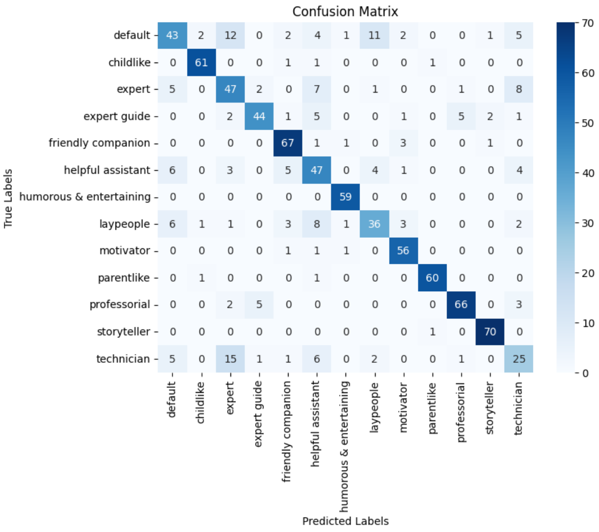


Figure 7: Distributions of sentiment scores for ChatGPT's default personality.



	default	childlike	expert	expert guide	friendly companion	helpful assistant	humorous & entertaining	laypeople	motivator	parentlike	professorial	storyteller	technician
default	43	2	12	0	2	4	1	11	2	0	0	1	5
childlike	0	61	0	0	1	1	0	0	0	1	0	0	0
expert	5	0	47	2	0	7	0	1	0	0	1	0	8
expert guide	0	0	2	44	1	5	0	0	1	0	5	2	1
friendly companion	0	0	0	0	67	1	1	0	3	0	0	1	0
helpful assistant	6	0	3	0	5	47	0	4	1	0	0	0	4
humorous & entertaining	0	0	0	0	0	0	59	0	0	0	0	0	0
laypeople	6	1	1	0	3	8	1	36	3	0	0	0	2
motivator	0	0	0	0	1	1	1	0	56	0	0	0	0
parentlike	0	1	0	0	0	1	0	0	0	60	0	0	0
professorial	0	0	2	5	0	0	0	0	0	0	66	0	3
storyteller	0	0	0	0	0	0	0	0	0	1	0	70	0
technician	5	0	15	1	1	6	0	2	0	0	1	0	25

Figure 8: Confusion matrix of the personality classifier.

7 Conclusion

The analysis showed that the personality styles simulated by ChatGPT are largely in line with requirements and expectations and can be used in adaptive human-centric XAI interfaces. ChatGPT is able to generate texts of appropriate length with a number of facts adapted to the personality. A clear distinction could be made between professional/technical and more emotional/human-centered personalities, which is of great importance for adaptive human-centered XAI interfaces. The use of stopwords and the readability of the texts behave according to the personality styles. ChatGPT is able to create the appropriate sentiment of a text and words and phrases are used that match the personalities. This was shown by training and analysis of a personality classifier and application of SHAP explanations.

8 Limitations

While ChatGPT has demonstrated the ability to effectively replicate diverse personality styles in textual analysis, the congruence of these simulations with real human perception remains unestablished. In order to provide clarity on this issue, it is necessary to examine how the simulated personalities affect the individual. Also, whether the personality styles can positively influence the important attributes of XAI interfaces, including trust, fairness and transparency, must be shown in future studies.

References

Rudolf Franz Flesch. 1948. A new readability yardstick. *The Journal of Applied Psychology*, 32(3):221–233.

Denis J. Hilton. 1990. Conversational processes and causal explanation. *Psychological Bulletin*, 107:65–81.

Sophie F. Jentzsch, Sviatlana Höhn, and Nico Hochgeschwender. 2019. Conversational interfaces for explainable ai: A human-centred approach. In *Calvaresi, D., Najjar, A., Schumacher, M., Främling, K. (eds) Explainable, Transparent Autonomous Agents and Multi-Agent Systems. EXTRAAMAS 2019. Lecture Notes in Computer Science()*, volume 11763, pages 77–92, Cham. Springer International Publishing.

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.

Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023. Evaluating ChatGPT's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness. *Computing Research Repository, arXiv:2304.11633. Version 1.*

Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4768–4777, Red Hook, NY, USA. Curran Associates Inc.

Philip M. McCarthy and Scott Jarvis. 2010. MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior Research Methods*, 42(2):381–392.

Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

Lorenz Mindner, Tim Schlippe, and Kristina Schaaff. 2023. Classification of human- and AI-generated texts: Investigating features for ChatGPT. In *Artificial Intelligence in Education Technologies: New Development and Innovative Practices*, pages 152–170, Singapore. Springer Nature Singapore.

Sandra Mitrović, Davide Andreoletti, and Omran Ayoub. 2023. ChatGPT or human? Detect and explain. Explaining decisions of machine learning model for detecting short ChatGPT-generated text. *Computing Research Repository, arXiv:2301.13852. Version 1.*

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *Computing Research Repository, arXiv:1910.01108. Version 4.*

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.

Mildred C. Templin. 1957. *Certain Language Skills in Children: Their Development and Interrelationships*, NED - New edition edition, volume 26. University of Minnesota Press.

Joan Torruella and Ramon Capsada. 2013. Lexical statistics and tipological structures: A measure of lexical richness. *Procedia - Social and Behavioral Sciences*, 95:447–454.

Thorsten Zylowski. 2022. Study on criteria for explainable AI for laypeople. In *Proceedings of the Second International Workshop on Explainable and Interpretable Machine Learning (XI-ML 2022) co-located with the 45rd German Conference on Artificial Intelligence (KI 2022)*, Trier (Virtual), Germany. CEUR Workshop Proceedings.

Topically diversified summarization of customer reviews

Florian Carichon and **Gilles Gaporossi**
HEC Montréal
3000, chemin de la Côte-Sainte-Catherine, Montréal (Québec), Canada, H3T 2A7.
florian.carichon@hec.ca
gilles.caporossi@hec.ca

Abstract

Promoting information coverage and sentence diversity is an efficient method to handle the fundamental issue of data heterogeneity or redundancy in multi-document summarization. We introduce a self-supervised algorithm for multi-document summarization that employs a multitask learning approach for topic diversification. Our model is based on two variational autoencoders that combine the training of a language model and a topic model to bias text generation and control the topic content of the produced summaries. We evaluate our method on the Amazon product review dataset and report ROUGE results and other metrics to assess information coverage. We demonstrate that our approach creates diversified outputs for the same batch of reviews and aspect-focused ones, allowing us to optimize text generation strategies.

1 Introduction

E-commerce and online sales platforms have grown substantially among the leading shopping media [1]. They change how we purchase products or services, allowing access to user experience. However, due to the subjective nature of reviews, customers must read many reviews to make an informed decision. By distilling the most important content in a reduced version of all opinions, automatic text summarization becomes crucial to help users.

The recent success of deep learning systems has led to significant improvement of extractive (Angelidis et al., 2021) or abstractive (See et al., 2017a; Paulus et al., 2017) document summarization models. With the domain-sensitive nature of product reviews, manufacturing large parallel corpora becomes costly and hardly transferable. Therefore, it has created a strong appetite for unsupervised summarization approaches where salient information depicts the consensual customer's point of view. However, the data heterogeneity of opinions distorts relevant content, resulting in overly broad summaries (Amplayo et al., 2021). Thus, It is essential to design strategies focusing on specific product aspects and transcribing this fine-grained content into the summary (Coavoux et al., 2019).

Since aspects can be implicitly grouped together according to themes, the detection of product review topics has naturally been associated with review aspects (Zhai et al., 2015). Methods such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and its deep learning variants have proven to be efficient in dynamically identifying these themes for opinion datasets (Ozyurt and Akcayol, 2021). In the context of opinion summarization, topic diversity increases the volume of the semantic space, improves information coverage, and therefore satisfies different needs of the user's population (Yogatama et al., 2015). The objective is then to optimize the sentences' topic relevancy and diversity (Li et al., 2010; Fang et al., 2015). Conditional variational autoencoders (Sohn et al., 2015) trained with topic modelling systems (Gao and Ren, 2019; Xiao et al., 2018) thus represent a promising avenue in this context.

In this article, we introduce an abstractive method for unsupervised customer opinion summarization that can produce text segments focused on various topics and combine them to maximize the input coverage. More specifically, our approach relies on a multi-task learning algorithm to train a topic and a language model jointly, both based on a variational autoencoder (VAE). We use the topic latent representation to condition the language model when learning review reconstruction. During the generation phase, we can select a subset of different topics to bias content included in the summary. We evaluated our approach on the Amazon product dataset, showing the importance of topic modelling to bring detailed and meaningful messages in such a heterogeneous context.

[1] https://www.forbes.com/advisor/business/ecommerce-statistics/

2 Related work

2.1 Multidocument Summarization for Opinion

Recent unsupervised abstractive techniques encapsulate information redundancy from a group of reviews into an average latent representation either directly (Chu and Liu, 2019; Bražinskas et al., 2020). However, such models suffer from aspects and topic heterogeneity, thus resulting in overly broad and almost irrelevant summaries. To address this issue, authors in (Angelidis and Lapata, 2018) create aspects-based representations with a partial autoencoder and devise an optimization function to select opinion that leverages their coverage. OpinionDigest (Suhara et al., 2020) is another method that clusters topically related reviews and employs a ranking algorithm to increase diversity in the output. Finally, (Amplayo et al., 2021) have introduced an interesting hybrid procedure that clusters opinions and extracts sentences to produce a summary predicated on popular or specific aspects. Regarding abstractive summarization, authors in (Coavoux et al., 2019) combine Meansum (Chu and Liu, 2019) with a clustering algorithm to conceive a latent representation for each group and form a text that maximizes input coverage. Our model is closely related because we modify the hierarchical VAE submitted in (Bražinskas et al., 2020) with a topic model. However, we propose a multi-task learning objective to produce dynamic topic representations, letting us condition the summary on the popular or specific topic/aspect.

2.2 Topic modeling

One of the most known and employed models is the Latent Dirichlet Allocation (LDA) (Blei et al., 2003) because of its generative ability and interpretability. The method has been applied in (Arora and Ravindran, 2008) for extractive summarization by submitting an algorithm that selects phrases with the highest probability of being produced by both the main topic and the document collection. Some authors have proposed increasing the coverage of the input texts by weighing the importance of the LDA topics with their similarity to ensure the diversity of sentences (Ren and de Rijke, 2015). Regarding recent deep learning models, some authors have adapted the re-parametrization trick of VAEs (Kingma and Welling, 2022) to multinomial distributions such as the Dirichlet distribution to create deep topic models (Srivastava and Sutton, 2017). Thereafter, such techniques have been used to obtain conditional language models to diversify sentence outputs. The idea is to produce biased latent representations by weighting input information by topics (Gao and Ren, 2019) or to concatenate directly the latent and the topic vectors (Xiao et al., 2018). Our approach combines these principles to learn relevant topics and optimize their selection for increasing opinion coverage in abstractive summarization.

3 Proposed Model

This section presents the general architecture of our multi-task learning approach as described in the figure 1. We first modify the hierarchical VAE summarizer proposed in (Bražinskas et al., 2020) by adding another VAE for topic modelling. We also introduce methods to select and condition summary generation regarding various topics.

The corpus is composed of customer reviews on different products. The vocabulary of the corpus is noted V. We define a batch of M customer reviews regarding a specific product as $\{R_1, ..., R_i, ..., R_M\}$ used to train our model. Each review R_i is composed of a set of words $X = \{X_1, ..., X_j, ..., X_N\}$, where N represents each review's variable length.

3.0.1 Topic Model

For a given review R_i, we apply a Bag of Words (BoW) encoding to obtain a vector BoW_i of size $|V|$, where dimensions indicate the word occurrence in R_i. This vector is then fed to a two-layer Forward Neural Network with a softplus activation function to create h_i^{bow}. We use this dense representation to encode the topic distribution through the continuous latent representation t_i. The objective of the model is to maximize the following:

$$\log \int \prod_{i=1}^{M} p_\theta(BoW_i|t_i, \beta) \quad (1)$$

where β represents the multinomial prior distribution of the topics over the vocabulary. As for the *ProdLDA* model (Srivastava and Sutton, 2017), we approximate the mixture of two multinomial distributions to their weighted multiplication. Therefore, we combine β and t_i to compute the probability of generating the output Bag of Words $BoW_i^{'}$:

$$p_\theta(BoW_i^{'}) = softmax([t_i \cdot \beta]) \quad (2)$$

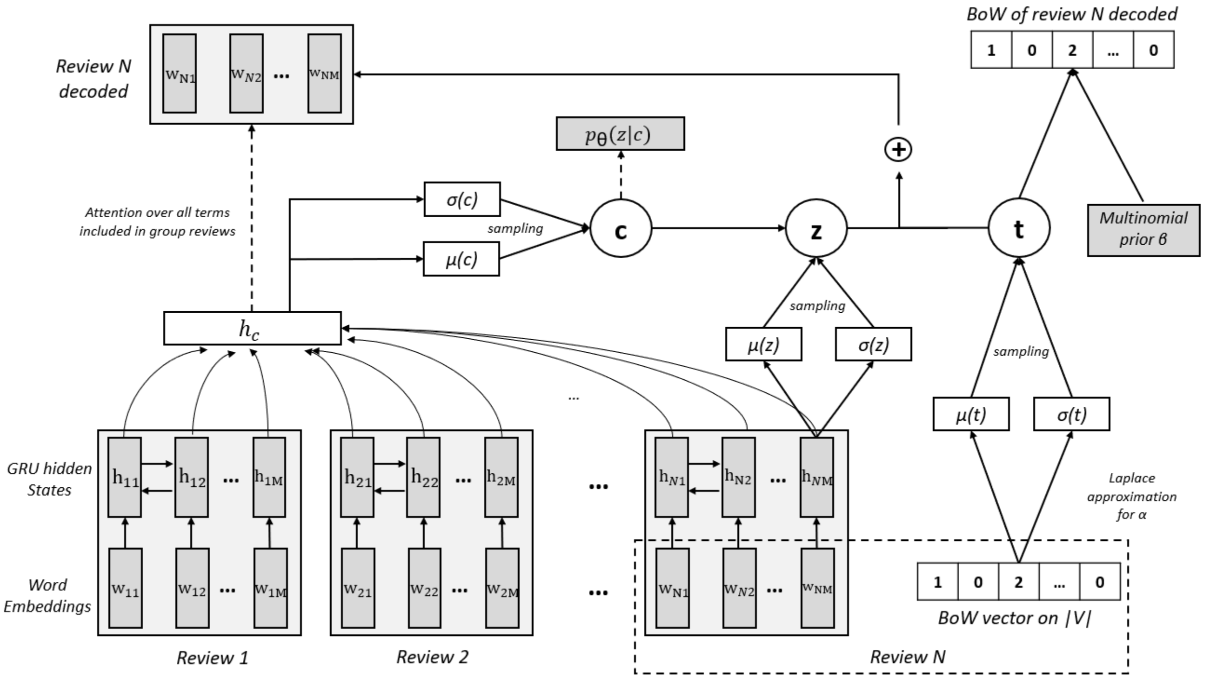


Figure 1: The multitask architecture for topic diversification of summary generation. The right part presents how the VAE is trained with a bag of word representation to obtain the topic distribution and the latent variable t. The left part displays the language model VAE. The latent variable c encodes the whole group of reviews while z encodes individual information. z is conditioned by c in training and is combined with t for the text reconstruction.

We train this part of the model with the mean square error function.

3.0.2 Language Model

We transform every input review with a pre-trained embedding model. The embedding matrix is fed to our encoder, a bidirectional Gated Recurrent Unit (GRU) (Cho et al., 2014). It produces an encoding h_{ij} for each word $j \in R_i$ and the last hidden state output h_{iN} used as the sentence representation. We based the training of our language model on the hierarchical VAE structure proposed in *Lsumm* (Bražinskas et al., 2020). Therefore, we first create the hidden representation h_c for all the group by computing the weighted sum over the attention of $m_{ij} = [h_{ij}; E_{ij}]$, the concatenation of the embedding and the GRU representations of the term x_j in R_i. We also assume a standard Gaussian distribution and apply a linear projection on h_c to sample the latent representation c encapsulating the information from the batch of reviews. Then, to perform the text reconstruction, we concatenate h_{iN}, the last GRU layer of R_i, and c to sample the latent variable z and pass it to our decoder.

When reconstructing, we perform N decoding steps to generate our sentence. We set the initial hidden state of the decoder, a simple GRU, s_0 to $[z_i; t_i]$ the concatenation of the topic and latent representation of R_i. At each decoding step t, we estimate the current hidden state s_t with the previous states s_{t-1} and predicted word $x^{'}_{t-1}$. We keep following the structure introduced in (Bražinskas et al., 2020) by calculating the attention distribution a^t, as in (Bahdanau et al., 2016), over the whole group of reviews $R_{\_i}$, excluding R_i. Once computed, we use every attention value $a^t_{\_i}$ to weight the representation $h_{\_i}$ of terms not belonging to R_i to create the context vector c_t. This vector is concatenated with the decoder state s_t and passed through a linear and a softmax layer to determine the probability of generating the output word $p_g(x^{'}_t)$:

$$P_g(x^{'}_t) = softmax(V^{'}(V[s_t, c^t] + b) + b^{'}) \tag{3}$$

where $V^{'}$, V, b, and $b^{'}$ are learnable parameters. We finally deploy a copy mechanism as presented in the *Pointer Generator Model* (PGN) (See et al., 2017b) to consider Out-Of-Vocabulary words. We compute the probability p_{gen} with a forward network and a sigmoid function over the context vector c_t, the hidden state s_t, and the previous predicted word $x_{t-1}^{'}$. The model uses p_{gen} to decide if it must preserve $x_t^{'}$ or to copy a term from R_i. The new probability then becomes:

$$P(x_t^{'}) = p_{gen} \times P_g(x_t^{'}) + (1 - p_{gen}) \times \sum_{i \in V_{ext}} (a^t_{\_i}) \tag{4}$$

where V_{ext} is the extended vocabulary aggregating the training vocabulary and the source document distribution. Finally, we let the model choose to draw terms directly from the distribution of topics $p(BoW_i^{'})$ defined in equation 3.0.1 by modifying the final probability:

$$P_{final}(x_t^{'}) = P(x_t^{'}) + p(BoW_i^{'}) \tag{5}$$

We empirically notice that letting the model choose between the two probabilities helps the model to converge better when learning the topic distribution. The language model is trained with the cross-entropy function.

3.0.3 General Architecture

Our complete approach combines the topic and the language models. The objective is to maximize the following function:

$$\log \int \left[p_\theta(c) \prod_{i=1}^{M} \int p_\theta(R_i | z_i, R\_i, BoW_i, t_i) \right. \\ \left. p_\theta(z_i|c) dz_i \right] dc + \log \int \prod_{i=1}^{M} p_\theta(BoW_i | t_i, \beta) dt_i \tag{6}$$

The right part of the function describes the topic model, and the left part depicts our language model conditioned by the topic content. This approach enables the system to learn relevant topics and use them to condition summary generation.

3.1 Model distributions

This section describes the assumptions about the prior and posterior distributions. We rely on the principles defined in (Bražinskas et al., 2020) for approximating c and z and (Srivastava and Sutton, 2017) for t. We refer the lecturer to these articles for further mathematical details.

3.1.1 Reconstruction latent variables: c ans z

For c, we assume a standard normal prior distribution $p(c) = \mathcal{N}(c; 0, I)$. For the posterior distribution, we use the reparameterization trick (Kingma and Welling, 2022) for Gaussian distribution with a linear projection on h_c. We estimate the mean $\mu_\Phi(c)$ and variance $\sigma_\Phi(c)$ the approximated inference network $q_\Phi(c|h_c) = \mathcal{N}(c; \mu_\Phi(h_c), I\sigma_\Phi(h_c))$. Regarding z, we also assume a prior normal Gaussian distribution. The major difference is that the latter is conditioned by c to obtain $p_\theta(z|c) = N(z; \mu_\theta(c), I\sigma_\theta(c))$. As for the mean $\mu_\Phi(z)$ and the variance $\sigma_\Phi(z)$ of the inference posterior distribution, we use the same procedure by linearly projecting the concatenation $[R_i; c]$. Then, we sample z through $q_\Phi(z_i|R_i, c) = N(z_i; \mu_\Phi(R_i, c), I\sigma_\Phi(R_i, c))$.

3.1.2 Topic latent variable: t

We assume a Dirichlet prior distribution for the latent topic variable t because it has been shown beneficial to obtain good and interpretable topics (Blei et al., 2003). The reparameterization trick becomes a Laplace approximation with a softmax estimation to compute the distribution and make it tractable within the VAE framework. This approximation to the topic prior $p_\theta(t|\alpha)$ is equivalent to considering a logistic normal distribution with parameters with mean $\mu_\theta(t)$ and covariance matrix $\sigma_\theta(t)$ that are functions of α and K the number of defined topics. Once we assume this distribution, we can once again compute the parameters of the posterior distribution from an inference network as a linear projection on h_i^{BoW} to obtain $q_\Phi(t_i|h_i^{BoW}) = N(t_i; \mu_\Phi(h_i^{BoW}), I\sigma_\Phi(h_i^{BoW}))$.

3.2 Model loss function

We seek to maximize the Evidence Lower BOund (ELBO) for variational inference regarding the parameters θ and Φ. The following equations depict the language model noted $\mathcal{L}_{LM}$ and the topic model loss $\mathcal{L}_{TM}$.

$$\mathcal{L}_{LM}(\theta, \Phi) = \mathbb{E}_{q_\Phi(c|R)} \left[\sum_{i=1}^{M} \mathbb{E}_{q_\Phi(z_i|R_i,c)} \right. \\ [\log p_\theta(R_i|z_i, t_i, BoW_i)] - \\ \left. \sum_{i}^{M} \mathbb{D}_{KL} \left[q_\Phi(z_i|R_i, c) || p_\theta(z_i|c) \right] \right] \\ - \mathbb{D}_{KL} \left[q_\Phi(c|R) || p_\theta(c) \right] \tag{7}$$

$$\mathcal{L}_{TM}(\theta, \Phi) = \sum_{i=1}^{M} \mathbb{E}_{q_\Phi(t_i|BoW_i)} [\log p_\theta(BoW_i|t_i, \beta) \\ -\mathbb{D}_{KL} [q_\Phi(t_i|BoW_i)||p_\theta(t_i|\alpha)]] \quad (8)$$

For both losses, the left part of the expressions ensures the text reconstruction of R_i or its bag of words representation BoW_i. The right term is the *Kullback-Leibler* divergence, which guarantees to match our prior distributions. We then minimize the joint loss as the sum of $\mathcal{L}_{LM}$ and $\mathcal{L}_{TM}$.

3.3 Summary Generation

To condition summary generation, we must first set up a strategy to designate the $k = [1, ..., K]$ main theme(s) on which to focus. We determine the relevant topics by identifying the ones that deviate the most from their expected prior distribution (AlSumait et al., 2009). To ensure further their diversity, we have implemented a Maximum Margin Relevance approach (Carbonell and Goldstein, 1998). Therefore, we choose topics from the posterior distribution that maximize $cos(t_k^{prior}, t_k) - \lambda * cos(t_k, t_j)$, where t_k is the topic distribution over our documents, t_j are the already picked ones, cos is the cosine similarity, and $\lambda = 0.5$.

For each selected topic k, we bias the hidden representation h_c with the posterior topic-word distribution β_k. We establish the set $X_{topics}k$ by preserving $1/8$ of the most topically probable terms in β_k from the extended vocabulary. We tested multiple filtering factors ranging from $1/2$ to $1/32$. Our first observations let us think that if we keep too many words, we do not impose enough diversity in the outputs, and if we remove too much, sentences become ungrammatical. Therefore, we empirically chose to preserve $1/8$ words as a good balance between the produced summaries' diversity and coherency. When creating h_c, instead of attending to all the group reviews' words, we attend only to $X_{topics}k$; the remaining words are masked. Then, we fix c to $\mu_\Phi(c)$ constructed via the inference model through this biased h_c.

To further condition the summarization of our text collection, we use the topic distribution t_k to set z to $z^{topic} = \mu_\theta(z) * t_k$, a topically biased representation of its prior mean for each document. We sample our summary by maximizing the probability expectation $P(x_t^{'})$ only. We instead apply $p(BoW_i^{'})$ in the beam search method to select among our K best-generated hypotheses the one that maximizes the sum of the two probabilities.

4 Experiment

4.1 Dataset

We trained our model on the Amazon Product dataset composed of reviews on 29 product categories (He and McAuley, 2016). We have considered products with at least 15 and a maximum of 100 reviews. We excluded texts under 8 and above 200 tokens. We remove the ones above the 90^{th} percentile each time. Since we aim to demonstrate the model's ability to handle heterogeneous information, we sample reviews from 19 categories and evaluate the model on the same 200 human-generated summaries as in (Bražinskas et al., 2020). Our final training data is composed of 17,497 reviews drawn from 303 products and the validation of 3,105 reviews from 50 products.

4.2 Implementation details

Our model uses the GloVe 200-dimensional pre-trained word embeddings (Pennington et al., 2014). The text was lowercased, and we used Spacy tokenizer and part-of-speech tagger [2] to preserve only adverbs, adjectives and nouns for the BOW representation. Both the model's encoder and decoder are composed of a single bidirectional layer with a size of 512 hidden units. We set the dimensions of the latent variable z and c to 600. We set the number of topics t to 30. We initialize the model during training with a Xavier uniform distribution (Glorot and Bengio, 2010). We trained the model for 150 epochs with the Adam optimizer (Kingma and Ba, 2017), a learning rate of $5 * 10^{-4}$, a weight decay of 10^{-6}, a gradient clipping of 10, and a dropout ration of 0.2. Regarding the KL divergence terms, we have employed a cycling function with $r = 0.8$ (Fu et al., 2019) and a maximum value of 1 for z and 0.65 for c. We have used a linear scheduling function between epochs 0 to 40 with a max value set to 1 for t. Finally, we apply the beam search method with a beam size established to 5 and an n-gram blocking method (Paulus et al., 2017) set to avoid trigram repetitions. Our code is available

[2] https://spacy.io/

on GitHub [3].

4.3 Evaluation

We compare our approach with 3 different baselines. The first is *BERT for Text Summarization* (Miller, 2019), and the second is *TextRank* (Mihalcea and Tarau, 2004). These two extractive methods are regularly used as baselines for evaluating general-purpose summarization. We also compare to our unsupervised base abstractive model *Lsumm* (Bražinskas et al., 2020). We trained and fine-tuned it on our Amazon dataset with the same parameters detailed in the section 4.2.

We report the average and maximum ROUGE F1 scores (Lin, 2004) for the different baselines on the evaluation dataset, which encompasses 3 human-created summaries for 60 products consisting of 8 reviews. We also provide the ROUGE scores with filtered stop words to emphasize the presence of content words in the generated outputs. We further include BLEURT scores (Sellam et al., 2020) to indicate to what extent the summaries convey the meaning of the input. Finally, we disclose how well methods can capture the topics addressed in the opinions expressed. To that extent, we train a LDA model with the Gensim library [4] on our training dataset. We then measure the similarity of the topic distributions and the semantic coherence of topics as described in (Greene et al., 2014) between the input reviews and the produced summaries.

5 Results and analysis

5.1 Model evaluation

We introduce two models based on our approach. The first method, *TopiCatSumm*, generates one summary based on K topically conditioned sentences of length N_{mean}/K, where N_{mean} is the average length of a batch of reviews. Since $N_{mean} = 58$ words, we set $K = 3$ to ensure diversity while generating long enough texts to be coherent. For the second, *TopicNSumm*, we have duplicated the evaluation dataset by making 3 topically distinct outputs of length N_{mean}. We report in the table 1 both the average score between the summary and all the references and the maximum score with its best matching reference. In the case of our second configuration *TopicNSumm*, we first pair each human production with the summary that optimizes its ROUGE score, and we report the average and maximum for all associated metrics.

Contrary to previous observations, self-supervised abstractive approaches appear worse than unsupervised extractive ones. This result likely represents the issues created by increasing data heterogeneity in the training set. The results also show that *TopicNSumm* allows an efficient optimization for matching related summaries with their reference. *TopiCatSumm* was the least performing, partly due to the size constraint penalizing the production of coherent sequences, but both methods improve the topic diversity and content coverage. We exhibit further these observations in the table 2.

These results reveal that both our approaches significantly improve content coverage and the topic distribution of the original customer opinions compared to the base abstractive approach. The filtered ROUGE scores further emphasize that our methods improve the ability to generate meaningful material. We assume that the performance of extractive strategies remains high because of the intrinsic homogeneity of a batch dealing with the same product. Therefore, we conduct a quick analysis of ROUGE for a batch of 16 reviews from 2 distinct products. Once again, we pair the best matching results to the summary to disclose average and maximum scores in the table 3. Our approaches suffer less from increasing heterogeneity, especially compared to extractive approaches, where the drop is the most important. In future studies, We plan to evaluate our model's capability to handle these extreme cases.

Finally, we provide some examples of generated documents by our model and the various baselines in appendix Appendix A..

5.2 Model and configuration analysis

We integrated our topic model with our language and summarization model during the training stage. We used a BOW vector for each review in our approach, but we could have created one for the entire group instead. However, by doing so, we observe that the model is unable to optimize both $\mathcal{L}_{TM}$ and $\mathcal{L}_{LM}$ at the same time. The need to capture individual and group information either restricts too much or brings too much noise into the latent variable t, penalizing the language or the topic model. During training, we also directly

[3] https://github.com/fcarichon/TopicDiversifiedVAESumm

[4] https://radimrehurek.com/gensim/models/ldamodel.html

Table 1: ROUGE scores on the Amazon dataset

Methods	R-1 (avg)	R-1 (max)	R-2 (avg)	R-2 (max)	R-L (avg)	R-L (max)
BERT Summarizer	25.03	30.33	4.17	7.39	15.31	18.67
TextRank	29.42	34.87	5.1	8.36	16.82	20.17
LSumm	17.57	21.92	0.51	1.14	10.91	13.59
TopiCatSumm	16.91	20.32	0.34	8.83	9.75	11.79
TopicNSumm	19.64	23.24	0.78	1.9	11.58	13.9

Table 2: Topic content and coverage evaluation on Amazon dataset

Methods	R-1 filt. (avg)	R-1 filt. (max)	BLEURT	Word topic overlap	topic similarity
Human references	NA	NA	-0.464	0.95	0.488
BERT Summarizer	18.64	25.04	-0.774	0.469	0.336
TextRank	21.24	27.29	-0.673	0.521	0.338
LSumm	6.67	9.89	-0.889	0.201	0.2
TopiCatSumm	9.39	12.71	-0,579	0.494	0.383
TopicNSumm	11.58	15.53	-0,677	0.678	0.477

concatenate z and t, but it would be tempting to do it for c and t since we use this representation as a condition for the summary generation. However, it results again in a significant drop for both losses, thus in low topic quality and inability to create diverse outputs. Including t in early layers makes it possible for the model to bypass learning the topic distribution, and it does not facilitate the task of capturing information for z as observed in (Xiao et al., 2018).

We have tested several configurations to bias the summary topically. The first experiments and the study of the output texts have emphasized the importance of employing the posterior distribution for sampling t and β matrices. With the data heterogeneity, using prior distributions leads to inducing broad topics, thus decreasing content quality and increasing hallucinations. The remaining iterative tests modify parameters in our current setup, stated as configuration 0 hereafter and described in section 3.3.

- Configuration 1: As for c and t, we have set t at its mean $mu(t)$ only.
- Configuration 2: We have tried to bias c with the main topic distribution by creating $c^{topic} = mu_{\Phi}(c) * t_k$ as we do for z^{topic}.
- Configuration 3: Inversely, instead of employing a topic biased z^{topic}, we have set it to its mean $z = mu_{\theta}(z)$ as in *Lsumm*.
- Configuration 4: Rather than masking the attention for h_c, we could weight the attention tensor with the word's topic probability.
- Configuration 5.a: Rather than masking attention at the group level, we have masked attention used directly in the decoder.
- Configuration 5.b: As for configuration 4, we have also tried to weight the decoder's attention tensor with the word's topic probability rather than masking it.
- Configuration 6: We have employed the BOW probability $p(BoW_i^{'})$ for the summary generation.

We report the ROUGE-1 and BLEURT results for the *TopicNSumm* model. We also provide a diversity metric to emphasize issues met by some configurations. To that end, we re-encode the generated summaries, and then measure the average cosine distance between these encodings. The table 4 displays the results obtained.

Results from configuration 1 emphasize again the value of having a precise and rich topic distribution to draw effectively relevant information from the topic distribution. The absence of difference in configuration 2 and the significant decrease of summaries' diversity in configuration 3 confirms the importance of biasing z as in training and not the group representation c, where the language model might compensate for the topic conditioning. The BLEURT and diversity scores of configuration 4 corroborate this hypothesis since implementing a soft bias, such as weighting the attention, is not enough to produce heterogeneous outputs. We can also note from analysis of configurations 5.a, 5.b, and 6 that directly impacting the text generation with topic distribution, in the decoder or the final probability distribution, is effective for producing relevant content. However, it comes at the expense of the summary coherency and readability.

Finally, another possibility is to let users bias the summary toward specific topics by defining

Table 3: Evaluation of the various approaches summarizing batches of 16 reviews sampled from 2 different products categories of the Amazon dataset.

Methods	R-1 (avg)	R-1 (max)	R-1 filt. (avg)	R-1 filt. (max)
BERT Summarizer	18.63	25.04	13.35	21.96
TextRank	21.24	27.29	14.02	23.58
LSumm	18.39	25.58	4.13	6.17
TopiCatSumm	16.67	22.17	9.11	15.17
TopicNSumm	18.45	25.15	11.04	18.02

Table 4: Table introducing the different results from various model configurations. We repeat the results of our main model in the first line for comparison.

TopicNSumm configurations	R-1 (avg)	R-1 (max)	BLEURT	Hidden diversity
Configuration 0	19.64	23.24	-0.677	0.578
Configuration 1	16.76	20.23	-0.656	0.513
Configuration 2	19.62	22.58	-0.69	0.534
Configuration 3	19.58	23.12	-0.65	0.328
Configuration 4	19.53	23.56	-0.72	0.469
Configuration 5.a	19.82	23.86	-0.63	0.557
Configuration 5.b	19.78	23.92	-0.61	0.558
Configuration 6	19.78	23.39	-0.677	0.562

their set of keywords $X^{user} = X_0^{user}, ..., X_U^{user}$. In that case, we identify the U main topics that maximize the probability $p(X^{user}|t_u)$ in the topic-word matrix. We provide 3 examples in table 7 in appendix Appendix B. of summaries generated by inputting the term “price” in the appendix. We observe that the model has conditioned the texts to include terms such as “expensive”, “full cost”, or even “budget”, which relate to the price. We also note that the model cannot bias the summary if the reviews do not deal with the input term. While this can be frustrating for the user, it is beneficial that the model does not hallucinate false information.

5.3 Limitations and future research avenues

The first limitation of our approach comes from the additional hyperparameters we introduced. We had to fine-tune many variables and distributions to make the model efficient. Specifically, we noticed that the number of topics selected is crucial since it influences the output quality and is, unfortunately, domain- or product-dependent. The second impediment of our method can be generalized to every system that tries to bias text generation. Indeed, biasing language models can lead to predicting terms that should not have been otherwise, inducing a potential loss of coherence or unwanted hallucinations. Finally, we are aware of the limitations of our architecture based on single-layer RNNs. The text coherency is inferior to current models predicated on pre-trained large language models (LLMs). Beyond the problems of budget and access to sufficiently powerful machines, studying simpler models guarantees that the capacity of these architectures does not absorb our approach and does induce diversity. We leave the analysis of its application to LLMs for future work.

6 Conclusion

In this paper, we introduced an unsupervised topic method for multi-document summarization of product reviews. It relies on two variational autoencoders combined in a multitask learning objective. This approach improves abstractive summarization models' performance by increasing content coverage or focusing on specific important topics. With this research, we hope that we have successfully demonstrated that this model could enhance the capacity of generative large language models to handle heterogeneous data and bias and diversify their outputs.

References

Loulwah AlSumait, Daniel Barbará, James Gentle, and Carlotta Domeniconi. 2009. Topic significance ranking of lda generative models. In *Machine Learning and Knowledge Discovery in Databases*, pages 67–82, Berlin, Heidelberg. Springer Berlin Heidelberg.

Reinald Kim Amplayo, Stefanos Angelidis, and Mirella Lapata. 2021. Aspect-controllable opinion summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6578–6593, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Stefanos Angelidis, Reinald Kim Amplayo, Yoshihiko Suhara, Xiaolan Wang, and Mirella Lapata. 2021. Extractive opinion summarization in quantized transformer spaces. *Transactions of the Association for Computational Linguistics*, 9:277–293.

Stefanos Angelidis and Mirella Lapata. 2018. Summarizing opinions: Aspect extraction meets sentiment prediction and they are both weakly supervised. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3675–3686, Brussels, Belgium. Association for Computational Linguistics.

Rachit Arora and Balaraman Ravindran. 2008. Latent dirichlet allocation based multi-document summarization. AND '08, page 91–97, New York, NY, USA. Association for Computing Machinery.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022.

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020. Unsupervised opinion summarization as copycat-review generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5151–5169, Online. Association for Computational Linguistics.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, page 335–336, New York, NY, USA. Association for Computing Machinery.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Eric Chu and Peter Liu. 2019. MeanSum: A neural model for unsupervised multi-document abstractive summarization. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1223–1232. PMLR.

Maximin Coavoux, Hady Elsahar, and Matthias Gallé. 2019. Unsupervised aspect-based multi-document abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 42–47, Hong Kong, China. Association for Computational Linguistics.

Hanyin Fang, Weiming Lu, Fei Wu, Yin Zhang, Xindi Shang, Jian Shao, and Yueting Zhuang. 2015. Topic aspect-oriented summarization via group selection. *Neurocomputing*, 149:1613–1619.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. 2019. Cyclical annealing schedule: A simple approach to mitigating KL vanishing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 240–250, Minneapolis, Minnesota. Association for Computational Linguistics.

Ce Gao and Jiangtao Ren. 2019. A topic-driven language model for learning to generate diverse sentences. *Neurocomputing*, 333:374–380.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy. PMLR.

Derek Greene, Derek O'Callaghan, and Pádraig Cunningham. 2014. How many topics? stability analysis for topic models. In *Machine Learning and Knowledge Discovery in Databases*, pages 498–513, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 507–517, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Diederik P Kingma and Max Welling. 2022. Auto-encoding variational bayes.

Xuan Li, Yi-Dong Shen, Liang Du, and Chen-Yan Xiong. 2010. Exploiting novelty, coverage and balance for topic-focused multi-document summarization. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, page 1765–1768, New York, NY, USA. Association for Computing Machinery.

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

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Derek Miller. 2019. Leveraging bert for extractive text summarization on lectures.

Baris Ozyurt and M. Ali Akcayol. 2021. A new topic modeling based approach for aspect extraction in aspect based sentiment analysis: Ss-lda. *Expert Systems with Applications*, 168:114231.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization.

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

Zhaochun Ren and Maarten de Rijke. 2015. Summarizing contrastive themes via hierarchical nonparametric processes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 93–102, New York, NY, USA. Association for Computing Machinery.

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

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

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models.

Yoshihiko Suhara, Xiaolan Wang, Stefanos Angelidis, and Wang-Chiew Tan. 2020. OpinionDigest: A simple framework for opinion summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5789–5798, Online. Association for Computational Linguistics.

Yijun Xiao, Tiancheng Zhao, and William Yang Wang. 2018. Dirichlet variational autoencoder for text modeling.

Dani Yogatama, Fei Liu, and Noah A. Smith. 2015. Extractive summarization by maximizing semantic volume. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1961–1966, Lisbon, Portugal. Association for Computational Linguistics.

ChengXiang Zhai, William W. Cohen, and John Lafferty. 2015. Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. *SIGIR Forum*, 49(1):2–9.

Appendix A. Models' generated texts

The table 5 presents the produced results by the different models for batches of 8 reviews. We note the better coherence and quality of the extractive baseline. However, we can also observe for vacuum filter examples that our method generated texts on the eating system, the filters and their price, or the fans. It highlights the ability of our model to increase the coverage of the inputs' topics and aspects. The table 6 shows the generated texts for a batch of 16 documents. The benefit of our approach is even more obvious here when we see 2 summaries focusing on the vacuum and the other on the steamer. In contrast, our baselines cannot manage this information diversity and have a considerable loss of coherency and relevance.

Appendix B. Texts with an input keyword

The table 7 presents the produced summaries by our model when we provide an input term to bias the generation of the model. A summary is then generated for each given term. The products presented here are the ones used in the previous examples to allow output comparison with this new bias.

Table 5: Table with examples of generated texts. For each product we provide the text generated by our two configurations, the absractive model LSumm and the extractive model TextRank.

B0002U34HY (CHV1510 Vacuum filter)	Our model TopiCatSumm	Easy fix before expected not much monster filters but with regular use handles clean, seems sturdy. However this filter was difficult with product support, I read comparable CHV1510 on here as other. The dirty class hitting washable model construction of functionality CHV1510 ridiculous, quality functionality washable.
	Our model TopicNSumm summary 1	CHV1510 games was home from eating all i complained without such cool 3rd CHV1510 brand I and amount on them off position not one time with filter that are just guessing all color!
	Our model TopicNSumm summary 2	that said filter and cheaper on shipping as hair fast shipping here than what should is but for something changed after working. The filter holder showed that, what appears it properly had different place for filter like using generic brand at all!
	Our model TopicNSumm summary 3	For the fans mounted cold lights: positive copies filters the world has broken open when aid properly from CHV1510, so in some amounts source on wrench breaking during these are fantastic and I still recommend
	LSumm	it says harder. to install with filter as possible for filter! it takes some amounts. it seems too strong as opposed the original one of it and
	TextRank	This is the wrong filter if you are buying the CHV1510 Hand Vacuum. This item list listed with the vacuum – 'frequently bought together' with the Black & Decker CHV9608 9.6 Volt Cyclonic-Action Cordless DustBuster BUT this filter does NOT fit!

Table 5: Table with examples of generated texts (Continued)

B0013EQ20Y (Frye Boots)	Our model TopiCatSumm	it wish my face soft hat, the boots it cozy lifts up nice. Comfy ugg Frye perfectly residue inside comfortable stretchy amounted just what the doctor ordered from boots all. I served comfy, boot though sticks right but quickly to safety snug evenly over, all socks together is
	Our model TopicNSumm summary 1	Indeed an excellent product and most excellent boots base and nice as wide in between all sizes up. It needs enough for all occasion beware of adjustments such all over cameras during!
	Our model TopicNSumm summary 2	Indeed comfy! Securely packaged, the it too and I am wearing! it makes great for heavy use thick rooms but tough construction and comfort, sound nicely tasted Frye but
	Our model TopicNSumm summary 3	comfy boots has already hanging down set I wish where had them on fire if there have many on bugs like paper itself while having. Overall this pair work well
	LSumm	it seems so sturdy enough like that is. it seems more sturdy than expected to get them again and was worth to try them! it seems more comfortable! it seems better with
	TextRank	they can be a beast to get on, like any boot fit to last; once on, they are incredibly comfortable. With a 20year break from not wearing Frye it was a pleasant surprise the quality has stood the test of time.

Table 6: Table with examples of generated text by the various models for batches of 16 reviews sampled from 2 different products of two different categories.

B0002U34HY vacuum filter & B00006IUVM kitchen steamer	Our model TopicNSumm summary 1	quality filters not do any reviews and picture looks as usual but for decades material seems fine but great purchase and deliver quality packaged! yeah and trust with
	Our model TopicNSumm summary 2	quality filter for many light steamer washable rice brand steamer, although is just easy enough without sending to play using without issues until much sized goes steamer easy too steam rice for each nut only goes straight smoothly
	Our model TopicNSumm summary 3	ladies! steam it has superior points of shelves from there : do something that? this steamer gives all aspects go, some kind opened without wearing them into this. So in some reviews from dragon appeared steam as directed, received mine ripped rice vegetables today
	LSumm	the filter is just what i needed. i have a lot of the filter and the filter. is not the same as the original filter.. is a great deal. is a great deal. is a great deal. is a very a very a very
	TextRank	This is the wrong filter if you are buying the CHV1510 Hand Vacuum. Sometimes I use the steamer for just one vegetable, or for rice, but it’s really nice to have the separate basket.

Table 7: Examples of generated texts by our TopicNSumm where we input the word "price" to the model.

B0013EQ20Y (Frye Boots)	comfy noticeable! easy boots comfortable leather is inexpensive and wonderfully easy quality although is heavy as long to high although! instead i do wish that i have ordering it or worn on amazon.com since that it broke in two, only bought it 4 and times full cost
B00006IUVM (Kitchen Steamer)	updated hard 3 days! steam as use to force me rice is perfect with all customers at work budget is able with hesitant help at night supply store, too expensive than to sell items.
B0002U34HY (CHV1510 Vacuum filter)	CHV1510 filters is too and save dust the legs on top because occasionally leave volume under cycle i make sure look for washable filter or something. maybe it only keeps wet VF08

Extracting higher-order logic formulas from English sentences

Alexandre Rademaker
IBM Research and FGV/EMAp
alexrad@br.ibm.com

Guilherme Lima
IBM Research
guilherme.lima@ibm.com

Renato Cerqueira
IBM Research
rcerq@br.ibm.com

Abstract

We proposed a framework and its implementation as a Python library for converting English utterances into higher-order logic (HOL) formulas. HOL extends first-order logic and provides flexibility for representing natural language semantics. Our library uses a broad-coverage and robust HPSG grammar for English to produce minimal recursive semantics (MRS) structures. These open-source technologies from the DELPH-IN Consortium balance a rigorous linguistic grounding and compositionality with practical aspects for natural language processing applications. Finally, we evaluated our approach over SICK, a popular dataset for text entailment.

1 Introduction

Over the last decades of research on natural language processing (NLP) and computational linguistics (CL), many approaches were proposed for extracting the meaning of linguistic utterances into machine-understandable and unambiguous structures called meaning representations, a task called semantic parsing or semantic analysis (Jurafsky and Martin, 2023). More broadly, the construction and reasoning with meaning representations of natural language expressions are in the context of computational semantics.

This article presents MRS Logic, a library to translate English sentences into logical formulas. MRS Logic is based on methods and tools already extensively studied in the literature and presented in Section 2. Still, it presents some novelty in integrating these resources, our representation language, and how sizeable existing knowledge bases can be easily reused for language understanding.

Consider the ambiguous sentence from Example (1). MRS Logic elucidates all possible interpretations for it, formalizing them in higher-order logic (HOL) expressions. Figure 1 presents two interpretations. Section 3 describes our transformation.

(1) The oil company ensured no chemicals poisoned the river.

From the last paragraph, we can highlight one aspect of our approach: we embrace the ambiguity of natural language. Example (1) has 52 possible interpretations, each representable by a logical expression. Dealing with all possible interpretations and postponing pruning as much as possible may be required by knowledge-intense applications. In many cases, only after linking the linguistic elements to the non-linguistic knowledge of the world can one effectively establish the pragmatics or the speaker's meaning (Quine, 1960; Bender et al., 2015).

Our proposal contrasts with the dominant approach in NLP, where tools shift from explicit symbolic semantic representation to non-compositional and opaque representations such as vector embeddings. Avoiding any strong claim about the requirements for any system that aims at language understanding, we shared some concerns reported in (Mitchell, 2023; Bender et al., 2021) with purely language-model-based tools. Nevertheless, we envision combining large language models (LLM) with symbolic methods in NLP. For instance, extracting relevant common-sense facts from a vast collection of texts.

A well-known problem in the NLP/CL literature is the appropriate metrics for evaluating text understanding and, consequently, the adequacy of the semantic representation formalisms. (Condoravdi et al., 2003) made a case for considering the recognition of text entailment (RTE) between natural language utterances, now broadly considered not a sufficient criterion for language understanding. Still, it remains accepted as a minimal necessary criterion. With that in mind, we evaluate the proposals for semantic representations by measuring their performance on supporting entailment and contradiction detection between pairs of sentences. Section 4 discusses the performance of our system in a balanced subset of a well-known RTE dataset.

$$\begin{aligned}&\exists\, x_{10}, \_oil\_n\_1\; x_{10} \wedge (\exists\, x_{23}, \_river\_n\_of\; x_{23}\; \wedge (\exists\, x_6, (\exists\, e_9, compound\; e_9\; x_6\; x_{10}\; \wedge \_company\_n\_of\; x_6) \wedge \\ &\quad (\exists\, e_3, \_ensure\_v\_1\; e_3\; x_6\; (\forall x_{19}, \_chemical\_n\_1\; x_{19} \rightarrow \neg(\exists\, e_{24}, \_poison\_v\_1\; e_{24}\; x_{19}\; x_{23})))))\end{aligned} \tag{1}$$

$$\begin{aligned}&\exists\, x_{10}, \_oil\_n\_1\; x_{10} \wedge (\forall x_{19}, \_chemical\_n\_1\; x_{19} \rightarrow \neg(\exists\, x_{23}, \_river\_n\_of\; x_{23}\; \wedge (\exists\, x_6, (\exists\, e_9, \\ &\quad compound\; e8\; x_6\; x_{10}\; \wedge \_company\_n\_of\; x_6) \wedge (\exists\, e_{24}\; e_3, \_ensure\_v\_1\; e_3\; x_6\; (\_poison\_v\_1\; e_{24}\; x_{19}\; x_{23})))))\end{aligned} \tag{2}$$

Figure 1: Two possible logical formulas expressing the possible interpretations for Example (1).

In Section 5, we make some final remarks.

To sum up, our contributions are (1) a framework to produce logical expressions in HOL from English sentences, leveraging ERG grammar and related technologies from the DELPH-IN Consortium; [1] (2) a balanced subset of SICK corpus, sharing our results on the evaluation of our tool on that and some findings.

2 Background

Our library will be described in Section 3, but first, we must describe the technologies we reused and integrated.

The main component of MRS Logic is the English Resource Grammar (ERG) (Flickinger, 2000; Flickinger et al., 2000; Copestake and Flickinger, 2000). The English Resource Grammar is a broad-coverage, linguistically precise, general-purpose computational grammar under continuous development since 1994. It is implemented in the theoretical framework of Head-driven Phrase Structure Grammar (Pollard and Sag, 1994) where both morphosyntactic and semantic properties of English are expressed in a declarative format. Combined with specialized processing tools, it can map running English text to highly normalized representations of meaning called Minimal Recursion Semantics (MRS) (Copestake et al., 2005). ERG is developed as part of the international Deep Linguistic Processing with HPSG Initiative (DELPH-IN). It can be processed by several parsing and realization systems, including the LKB grammar engineering environment (Copestake, 2002), as well as more efficient parsers such as ACE (Crysmann and Packard, 2012).[2]

MRS structures are expressive and have a direct interface with syntax. It can be underspecified in many ways; here, we will describe the underspecification of fine-grained senses and quantifiers' scope. Underspecification allows a single MRS to capture a set of interpretations. Figure 2 shows one among the five possible MRSs for Example (1). It consists of a multiset of relations called elementary predications (EPs). An EP usually corresponds to a single lexeme but can represent grammatical features (e.g., `compound` and `udef_q`, called abstract predicates). Each EP has a label or handle, a predicate symbol, which, in the case of lexical predicates, encodes information about lemma, part-of-speech, and coarse-grained sense distinctions, and a list of numbered arguments: `ARG0`, `ARG1`, etc. The value of an argument can be either a scopal variable (a hole representing the places where alternative labels could fill) or a non-scopal variable (events, states, or entities). The `ARG0` argument has the EP's distinguished variable. This variable denotes an event, state, or referential or abstract entity (e_i or x_i, respectively). Each non-quantifier EP has its unique distinguished variable. Finally, an MRS has a set of handle constraints describing how the EPs' scopal arguments can be nested with EP labels. A constraint $h_i =_q h_j$ denotes equality modulo quantifier insertion. In addition to the indirect linking through handle constraints, EPs are directly linked by sharing the same variable as argument values, capturing the predicate-argument structure of the sentence. Finally, MRS also records properties on variables indicating morpho-syntactic marks of person, number, tense, aspect, etc.

In Figure 2, we see the MRS of the Example (1) where the topmost relation is `_ensure_v_1`, which has the non-empty arguments x_6 and h_{16}. The x_6 is the distinguished variable of the relation `_company_n_of`. A handle constraint equates the sentential variable h_{16} with h_{22}, the label of `_poison_v_1`. The rest of the EPs can be explained similarly. Note that h_5 does not appear in the handle constraints, suggesting that we have more than one possible way to equate this hole with the available labels.

The underspecification of scopes in the MRS of Figure 2 can be represented as the dominance graph (Koller and Thater, 2005) in Figure 3, a directed graph with two kinds of edges: tree edges and dominance edges (in red). Dominance graphs

[1] `https://github.com/delph-in/docs/wiki`

[2] `http://sweaglesw.org/linguistics/ace/`

$$\begin{array}{l}
\langle\, h_1, e_3, \\
\quad h_4{:}\text{\_the\_q}\langle 0{:}3\rangle(\text{ARG0 } x_6\{\text{PERS } \mathit{3}, \text{NUM } \mathit{sg}\}, \text{ RSTR } h_7, \text{ BODY } h_5), \\
\quad h_8{:}\text{compound}\langle 4{:}15\rangle(\text{ARG0 } e_9\{\text{SF } \mathit{prop}, \text{TENSE } \mathit{untensed}, \text{MOOD } \mathit{indicative}, \text{PROG -}, \text{PERF -}\}, \text{ ARG1 } x_6, \text{ ARG2 } x_{10}), \\
\quad h_{11}{:}\text{udef\_q}\langle 4{:}7\rangle(\text{ARG0 } x_{10}, \text{ RSTR } h_{13}, \text{ BODY } h_{12}), \\
\quad h_{14}{:}\text{\_oil\_n\_1}\langle 4{:}7\rangle(\text{ARG0 } x_{10}), \\
\quad h_8{:}\text{\_company\_n\_of}\langle 8{:}15\rangle(\text{ARG0 } x_6, \text{ ARG1 } i_{15}), \\
\quad h_2{:}\text{\_ensure\_v\_1}\langle 16{:}23\rangle(\text{ARG0 } e_3\{\text{SF } \mathit{prop}, \text{TENSE } \mathit{past}, \text{MOOD } \mathit{indicative}, \text{PROG -}, \text{PERF -}\}, \text{ ARG1 } x_6, \text{ ARG2 } h_{16}), \\
\quad h_{17}{:}\text{\_no\_q}\langle 24{:}26\rangle(\text{ARG0 } x_{19}\{\text{PERS } \mathit{3}, \text{NUM } \mathit{pl}, \text{IND } \mathit{+}\}, \text{ RSTR } h_{20}, \text{ BODY } h_{18}), \\
\quad h_{21}{:}\text{\_chemical\_n\_1}\langle 27{:}36\rangle(\text{ARG0 } x_{19}), \\
\quad h_{22}{:}\text{\_poison\_v\_1}\langle 37{:}45\rangle(\text{ARG0 } e_{24}\{\text{SF } \mathit{prop}, \text{TENSE } \mathit{past}, \text{MOOD } \mathit{indicative}, \text{PROG -}, \text{PERF -}\}, \text{ ARG1 } x_{19}, \text{ ARG2 } x_{23}), \\
\quad h_{25}{:}\text{\_the\_q}\langle 46{:}49\rangle(\text{ARG0 } x_{23}\{\text{PERS } \mathit{3}, \text{NUM } \mathit{sg}, \text{IND } \mathit{+}\}, \text{ RSTR } h_{27}, \text{ BODY } h_{26}), \\
\quad h_{28}{:}\text{\_river\_n\_of}\langle 50{:}55\rangle(\text{ARG0 } x_{23}, \text{ ARG1 } i_{29}) \\
\ \{\, h_1 =_q h_2,\ h_7 =_q h_8,\ h_{13} =_q h_{14},\ h_{16} =_q h_{22},\ h_{20} =_q h_{21},\ h_{27} =_q h_{28} \,\}\,\rangle
\end{array}$$

Figure 2: The first MRS return by ERG for the Example (1).

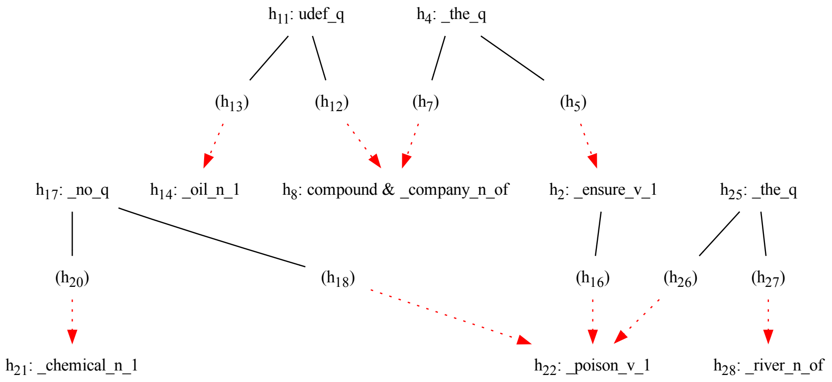


Figure 3: A dominance graph of the MRS from Figure 2.

are used as underspecified descriptions that can be solved to sets of scope trees [3] that later can be realized as formulas in some formal language. Figure 4 shows one of the 32 possible scope trees for the dominance graph from Figure 3. For computing the dominance graph and all possible scope trees for an MRS, we use Utool (Koller and Thater, 2005, 2006, 2010), a GUI and library written in Java.[4].

The scope trees are not directly useful for reasoning, and the CL literature has many proposals for representing NL utterance semantics. One of the most fundamental issues about which logic to use is whether one assumes any structure on the individuals. Other issues are the complexity, decidability, and tools for reasoning in a particular logic. So far, it is reasonable to accept that no existing logic is adequate for all the phenomena of natural language – although we acknowledge different logics individually capture some of the phenomena already studied.

Type theories are widely used in formal theories of the semantics of natural languages (Chatzikyriakidis and Luo, 2020; Ranta, 1994; Winter, 2016).

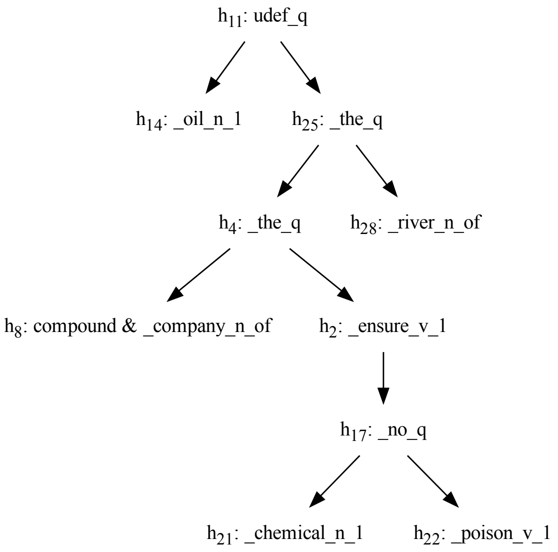


Figure 4: One possible scope tree resolved from the dominance graph from Figure 3.

[3] We are adopting the term suggested by (Emerson, 2020).

[4] https://github.com/coli-saar/utool/

A subset of that, simple type theory, also called higher-order logic (HOL), is a natural extension of first-order logic, which is elegant, highly expressive, and practical (Farmer, 2008). Inspired by modern implementations of simple type theory, such as HOL Light (Harrison, 2009) and Isabelle/HOL (Nipkow et al., 2002), and also by interactive proof assistants based on dependent type theory, such as Lean (Moura and Ullrich, 2021) and Coq (The Coq Development Team, 2021), we implemented the ULKB Logic (Lima et al., 2023). The formulas presented in Figure 1 are HOL formulas encoded in ULKB Logic. ULKB is an open-source framework written in Python for logical reasoning over knowledge graphs. It provides an interactive theorem prover-like environment that can interact with external provers like the E prover (Schulz et al., 2019) and the Z3 SMT solver (de Moura and Björner, 2008).

Finally, consider the possible senses for the word 'company.' ERG only distinguishes senses that are morphosyntactically marked. Since further sense distinctions could never be disambiguated based on grammatical structure alone, the ERG predicate symbol `_company_n_of` intended to be an underspecified representation of all the specific word senses. Wordnet 3.0 (Miller, 1995) contains nine possible nominal senses for this word. We use UKB (Agirre and Soroa, 2009) for Word Sense Disambiguation (WSD), the ERG predicates. It is a collection of programs for performing graph-based and lexical similarity using a pre-existing knowledge base.

3 MRS to Logic

MRS Logic is a library built on top of PyDelphin (Goodman, 2019) and ULKB.[5] It uses PyDelphin to coordinate the call to ERG and iterate over all possible MRS. An MRS is transformed into a scope tree using Utool and finally translated to ULKB formulas. MRS-Logic integrates all technologies described in Section 2. This section describes the translation of scope trees into ULKB formulas, skipping the implementation details of data structures and some design decisions.

At the high level, the translation starts from the topmost node of the scope tree, the handle in the higher position, usually a quantifier. The transformation sketched out in Figure 5 considered the scope tree from Figure 4 as input, and it works recursively.

The node h_{11} is the implicit quantifier udef_q,[6], as all other ERG quantifiers, it is modeled as a generalized (binary) quantifier (Westerståhl, 2019). We interpret this predicate as an existential quantifier in HOL. Nodes h_4 and h_{25} have the same interpretation but are surface predicates.[7] Node h_{17} is another quantifier; our current interpretation is as a universally quantified implication of the restriction to the negation of the body.

Note that variable x_{10} is instantiated, and further transformations of nodes h_{14} and h_{25} will be under the scope of this existential quantifier. Nodes h_{14}, h_{21} are trivial; ERG predicates are transformed into HOL predicates with the same arity. Node h_{28} has one uninstantiated parameter; the lexical entry for _river_n_of in ERG expects one optional complement.[8] Since the parameter was not supplied in the sentence, we decreased the cardinality of the generated HOL predicate. This behavior is configurable in our transformation and may be disabled if needed. The same simplification happens in transforming the predicate _company_n_of in h_8.

Node h_{22} has a verbal predicate with an event variable as its distinguished variable, `ARG0`. Event variables are not explicitly quantified in MRS, so we must decide when to introduce them in the HOL formula. The problem is that the existential quantifier for the event should not get a broad scope if negation is involved. Consider the sentence 'No man is walking' and a problematic translation to $\exists\, e2, \forall\, x3, \_man\_n\_1\ x3 \to \neg \_walk\_v\_1\ e2\ x3$. We don't want to instantiate e_2 to say later that it didn't exist. The correct approach is to instantiate the event variables as close as possible to the predicate with this variable as its distinguished variable.

Node h_2 is where HOL stands out. The verb 'ensure' can be taken as a factive verb (Hazlett, 2010) introducing a presupposition; that is, the HOL predicate gets a HOL formula as an argument, a higher-order construction not permitted in FOL.[9] In this example, this is the only case where $\mathbf{T}$ is applied recursively in a predicate argument.

[5] The code is available at `https://github.com/ibm/mrs-logic`.

[6] `https://github.com/delph-in/docs/wiki/ErgSemantics_ImplicitQuantifiers`

[7] `https://github.com/delph-in/docs/wiki/ErgSemantics_Basics`

[8] Consider 'the river of Colorado.'

[9] We acknowledge that FOL translations for the same phenomena are possible (Bos, 2014).

We haven't yet introduced in the system extra axioms to impose the presupposition reading when needed. Still, the translation presented here would not be affected by such additional axioms once we have a complete understanding of them.[10]

Finally, node h_8 is the only one with more than one EP; the transformation considers all EP with the same label as a coordination of the translations of each EP. We notice the ERG predicate compound; it can be considered an underspecified preposition. ERG analyses noun-noun compounds so that compound has the same structure as other explicit prepositions, e.g., 'boxes on tables are blue', 'boxes for tables are blue,' and 'table boxes are blue.'

The transformation creates the HOL predicates inline, but it could also pre-declared them as polymorphic predicates, such as $\_oil\_n\_1 : a \to bool$ where a is a type variable in ULKB.[11]

The translation covers some additional phenomena not illustrated in the example we used. Nevertheless, presenting it in the way we did gives the reader better intuition about its general ideas. We plan to extend our translation to all other ERG abstract predicates that model additional NL phenomena like normalization, disjunctions, conjunctions, etc.

4 SICK Experiment

The SICK dataset includes 9,840 sentence pairs taken from images and video captions. A selection of sentences from each source was used to produce pairs of sentences in 3 steps detailed reported in (Marelli et al., 2014; Bentivogli et al., 2016). The pairs were manually annotated regarding semantic similarity and logical relation: entailment, contradiction, or neutral.[12] The sentence pairs are rich in lexical, syntactic, and semantic phenomena. Still, the entailment test of the sentences was expected to be supported by common sense and grammatical knowledge and not to require encyclopedic knowledge about entities of the world.

Given the sentence's intended simplicity compared to previous datasets for Recognising Textual Entailment (RTE), SICK is excellent for testing our tool. Suppose our translation effectively captures the meaning of the sentences in HOL expressions. If sentences A and B are classified as an entailment, we should be able to prove $\Delta \vdash \mathbf{T}(A) \to \mathbf{T}(B)$ where Δ is a background theory.[13] If they are classified as a contradiction, we should be able to prove $\Delta \vdash \neg(\mathbf{T}(A) \wedge \mathbf{T}(B))$; otherwise, we consider them as neutral. This is a very simple approach compared to other logical-based RTE reports (Bos, 2014), but our goal here is to have a preliminary test of our transformation, not to improve the results on the SICK leaderboard.[14]

We start pre-parsing all sentences with ERG, asking for at most ten readings for each sentence. Of the 6,077 unique sentences, 3,435 sentences have the maximum number of readings we requested. 2,055 sentences had less than five readings, 564 between 5 and 9 readings, and only 23 sentences were not parsed by ERG, some of them by being ungrammatical (Kalouli et al., 2017a,b). This shows that we have a high degree of ambiguity, even for a collection of relatively simple sentences. Given that, during the main loop of the experiment, when we test each pair for logical entailment, we check at most four combinations of interpretations (HOL formulas) in a breath-first search strategy.

Unfortunately, SICK is very unbalanced regarding the entailment test, as we can see in Table 1, and the corpus contains a lot of repeated sentences. To overcome these limitations, we created a subset of SICK, called SB-SICK (small and balanced SICK), with 330 pairs for each label and no sentence repetition.

#	label
1424	CONTRADICTION
2822	ENTAILMENT
5596	NEUTRAL

Table 1: distribution of SICK sentences for each entailment label.

Table 2 summarizes the results we obtained. In this experiment, we did not use the WSD module, relying on the ERG predicates and its coarse-grained senses. The Δ is a small theory of 24 axioms we added incrementally during the tests, experimenting with the system's adaptabil-

[10] Note that presuppositions are one of many NL phenomena where the deep language processing with ERG, with a curated lexicon, and the kind of semantic analysis we are carrying on here makes the difference. For instance, if we take 'ensure' as a factive verb, we can adequately formalize its meaning. If an entity X ensures Y, Y should be taken as a true statement?

[11] A configuration can also specify a single type for all predicate parameters.

[12] We are only interested in the logical relations.

[13] The $\mathbf{T}(a)$ means the transformation of the NL sentence A into HOL formulas as described in Section 3.

[14] https://paperswithcode.com/dataset/sick

$$
\begin{aligned}
\mathbf{T}[h_{11}] &= \mathbf{T}[\mathsf{udef\_q\ ARG0}\ \ x_{10}\ \mathsf{RSTR}\ \ h_{14}\ \mathsf{BODY}\ \ h_{25}] = (\exists\, x_{10}, \mathbf{T}[h_{14}] \wedge \mathbf{T}[h_{25}])\\
\mathbf{T}[h_{14}] &= \mathbf{T}[\mathsf{\_oil\_n\_1\ ARG0}\ \ x_{10}] = \_oil\_n\_1(x_{10})\\
\mathbf{T}[h_{28}] &= \mathbf{T}[\mathsf{\_river\_n\_of\ ARG0}\ \ x_{23}\ \mathsf{ARG1}\ \ i_{29}] = \_river\_n\_of(x_{23})\\
\mathbf{T}[h_{21}] &= \mathbf{T}[\mathsf{\_chemical\_n\_1\ ARG0}\ \ x_{19}] = \_chemical(x_{19})\\
\mathbf{T}[h_{22}] &= \mathbf{T}[\mathsf{\_poison\_v\_1\ ARG0}\ \ e_{24}\ \mathsf{ARG1}\ \ x_{19}\ \mathsf{ARG2}\ \ x_{23}] = \exists\, e_{24}, \_poison\_v\_1(e_{24}, x_{19}, x_{23})\\
\mathbf{T}[h_{25}] &= \mathbf{T}[\mathsf{\_the\_q\ ARG0}\ \ x_{23}\ \mathsf{RSTR}\ \ h_4\ \mathsf{BODY}\ \ h_{28}] = (\exists\, x_{23}, \mathbf{T}[h_4] \wedge \mathbf{T}[h_{28}])\\
\mathbf{T}[h_4] &= \mathbf{T}[\mathsf{\_the\_q\ ARG0}\ \ x_6\ \mathsf{RSTR}\ \ h_8\ \mathsf{BODY}\ \ h_2] = (\exists\, x_6, \mathbf{T}[h_8] \wedge \mathbf{T}[h_2])\\
\mathbf{T}[h_8] &= \mathbf{T}[\mathsf{compound\ ARG0}\ \ e_9\ \mathsf{ARG1}\ \ x_6\ \mathsf{ARG2}\ \ x_{10}, \mathsf{\_company\_n\_of\ ARG0}\ \ x_6\ \mathsf{ARG1}\ \ i_{15}]\\
&= (\mathbf{T}[\mathsf{compound\ ARG0}\ \ e_9\ \mathsf{ARG1}\ \ x_6\ \mathsf{ARG2}\ \ x_{10}] \wedge \mathbf{T}[\mathsf{\_company\_n\_of\ ARG0}\ \ x_6\ \mathsf{ARG1}\ \ i_{15}])\\
&= (\exists\, e_9, compound(e_9, x_6, x_{10}) \wedge \_company\_n\_of(x_6))\\
\mathbf{T}[h_2] &= \mathbf{T}[\mathsf{\_ensure\_v\_1\ ARG0}\ \ e_3\ \mathsf{ARG1}\ \ x_6\ \mathsf{ARG2}\ \ h_{17}] = (\exists\, e_3, \_ensure\_v\_1(e_3, x_6, \mathbf{T}[h_{17}]))\\
\mathbf{T}[h_{17}] &= \mathbf{T}[\mathsf{\_no\_q\ ARG0}\ \ x_{19}\ \mathsf{RSTR}\ \ h_{21}\ \mathsf{BODY}\ \ h_{22}] = (\forall\, x_{19}, \mathbf{T}[h_{21}] \rightarrow \neg\mathbf{T}[h_{22}])
\end{aligned}
$$

Figure 5: The transformations of the MRS from Figure 4.

ity to incremental addition of background knowledge. The axioms cover simple lexical semantics gaps such as $\forall x, man\ x \rightarrow person\ x$ and $\forall x, empty\ x \rightarrow \neg full\ x$ that can be easily derived from resources like Wordnet. We also have some axioms related to the ERG abstract predicates, such as $\forall\ e\ x\ y, compound\ e\ x\ y \rightarrow for\ e\ x\ y$[15] and an axiom to deal with the null contribution to the semantics of expletive constructions, $\forall\ x\ y, \_be\_v\_there\ x\ y$.

label	true	false	%
CONTRADICTION	117	213	35
ENTAILMENT	132	198	40
NEUTRAL	330	-	100

Table 2: The results in the SB-SICK. The 'true' means that using MRS Logic, we proved the expected logical relation, and 'false' otherwise. Since neutral is a fallback in our method, we had no error for the neutral label.

We analyzed the cases where we could not prove the expected result, looking for possible translation failures. We summarized some relevant cases we found next, but none were related to problems with translating the MRSs to HOL.

As reported in (Kalouli et al., 2017a,b), we also found that some pairs have wrong labels. For instance, Examples (2) and (3) were annotated as entailment and contradiction, respectively. Example (2) is far from a logical entailment, although somehow related pragmatically. The SICK authors acknowledge these cases as inconsistencies in their dataset.[16]

(2) a. "People are walking outside the building that has several murals on it."
b. "Several people are in front of a colorful building."

(3) a. "The black and white dog is running indoors."
b. "The black and white dog is running in a green yard."

Errors in logical reasoning were also expected since we submitted the HOL formulas to FOL provers, relying on their ability to reduce them to FOL when possible.[17] We also have not implemented axioms to handle all ERG abstract predicates, e.g., nominalization. Some additional background knowledge is undoubtedly necessary and can eventually be induced (Ihsani, 2012), consider formalizing that a guitar player is a guitarist in Example (4).

(4) a. "A person has blonde and flyaway hair and is playing a guitar."
b. "A guitarist has blonde and flyaway hair."

We stressed our transformation rules without finding errors. Second, aligned (Bos, 2014), we validated that the lack of a systematic way to produce relevant background knowledge is the bottleneck of logical inference in RTE.

[15] Remember that compound means an underspecified preposition in the noun-noun compounds. This would be one of many axioms for each possible preposition in English.

[16] Notice that nothing blocks an interpretation of a situation with two distinct dogs or groups.

[17] Some high-order predicates, in the absence of explicit types, can be taken as functions.

5 Conclusion

We presented an open-source library to translate English sentences into HOL formulas. The code is available at http://github.com/ibm/mrs-logic. We tested the library in a dataset of pairs of sentences classified as entailment, contradiction, and neutral. Despite the results, we have collected the necessary insights to refine the RTE procedure and learned that a lot depends on the precise RTE task definition. Fine-grained deep linguistic analyses reveal inconsistencies invisible for purely statistical methods, hiding the real challenge of language understanding.

Considering the most popular approaches for RTE, we differ in using multiple interpretations for each sentence (although limited to four combinations) provided by the grammar-based analyses. We suspect that background knowledge is crucial for selecting the most plausible reading of the sentences when a pair is being tested. Many cases were not proved just because the expected readings of each sentence were not among the tested combinations limited by the computational resources we had.

The literature on computational semantics is vast. We are aware of the range of possibilities from non-compositional representations such as AMR (Banarescu et al., 2013) to inferences directly over surface forms such as Natural Logic (MacCartney and Manning, 2007). We focused on MRS, related to (Lien, 2014), although using logic inference instead of graph matching.

References

Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank for word sense disambiguation. In *The 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 33–41, Athens, Greece. ACL.

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 *The 7th linguistic annotation workshop and interoperability with discourse*, pages 178–186.

Emily M. Bender, Dan Flickinger, Stephan Oepen, Woodley Packard, and Ann Copestake. 2015. Layers of interpretation: On grammar and compositionality. In *The 11th International Conference on Computational Semantics*, pages 239–249, London, UK. ACL.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *The 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. ACM.

Luisa Bentivogli, Raffaella Bernardi, Marco Marelli, Stefano Menini, Marco Baroni, and Roberto Zamparelli. 2016. Sick through the semeval glasses. lesson learned from the evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. *Language Resources and Evaluation*, 50:95–124.

Johan Bos. 2014. Is there a place for logic in recognizing textual entailment. *Linguistic Issues in Language Technology*, 9.

Stergios Chatzikyriakidis and Zhaohui Luo. 2020. *Formal Semantics in Modern Type Theories*. Wiley.

Cleo Condoravdi, Dick Crouch, Valeria de Paiva, Reinhard Stolle, and Daniel G. Bobrow. 2003. Entailment, intensionality and text understanding. In *The HLT-NAACL 2003 Workshop on Text Meaning*, pages 38–45.

Ann Copestake. 2002. *Implementing typed feature structure grammars*, volume 110. CSLI publications Stanford.

Ann Copestake and Dan Flickinger. 2000. An open source grammar development environment and broad-coverage english grammar using hpsg. In *The Second Linguistic Resources and Evaluation Conference*, pages 591–600, Athens, Greece.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A. Sag. 2005. Minimal recursion semantics: An introduction. *Research on language and computation*, 3:281–332.

Berthold Crysmann and Woodley Packard. 2012. Towards efficient HPSG generation for German, a non-configurational language. In *COLING*, page 695–710.

Leonardo de Moura and Nikolaj Björner. 2008. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340, Berlin. Springer.

Guy Emerson. 2020. Linguists who use probabilistic models love them: Quantification in functional distributional semantics. In *The Probability and Meaning Conference (PaM 2020)*, pages 41–52, Gothenburg. ACL.

William M. Farmer. 2008. The seven virtues of simple type theory. *Journal of Applied Logic*, 6(3):267–286.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28.

Dan Flickinger, Ann Copestake, and Ivan A. Sag. 2000. Hpsg analysis of english. In *Verbmobil: Foundations of speech-to-speech translation*, pages 321–330. Springer, Berlin, Germany.

Michael Wayne Goodman. 2019. A python library for deep linguistic resources. In *2019 Pacific Neighborhood Consortium Annual Conference and Joint Meetings (PNC)*, Singapore.

John Harrison. 2009. HOL Light: An overview. In *Theorem Proving in Higher Order Logics*, pages 60–66, Berlin. Springer.

Allan Hazlett. 2010. The myth of factive verbs. *Philosophy and Phenomenological Research*, 80(3):497–522.

Annisa Ihsani. 2012. Automatic induction of background knowledge axioms for recognising textual entailment. Master's thesis, University of Groningen.

Daniel Jurafsky and James H. Martin. 2023. Speech and language processing. Draft of January 7, 2023.

Aikaterini-Lida Kalouli, Livy Real, and Valeria de Paiva. 2017a. Correcting contradictions. In *The Computing Natural Language Inference (CONLI)*, Montpellier, France.

Aikaterini-Lida Kalouli, Livy Real, and Valeria De Paiva. 2017b. Textual inference: getting logic from humans. In *The 12th International Conference on Computational Semantics (IWCS)*, Montpellier, France.

Alexander Koller and Stefan Thater. 2005. Efficient solving and exploration of scope ambiguities. In *The ACL Interactive Poster and Demonstration Sessions*, pages 9–12, Ann Arbor, Michigan. ACL.

Alexander Koller and Stefan Thater. 2006. An improved redundancy elimination algorithm for underspecified representations. In *The 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, pages 409–416, Sydney, Australia. ACL.

Alexander Koller and Stefan Thater. 2010. Computing weakest readings. In *The 48th Annual Meeting of the ACL*, pages 30–39, Uppsala, Sweden. ACL.

Elisabeth Lien. 2014. Using Minimal Recursion Semantics for entailment recognition. In *The Student Research Workshop at the 14th Conference of the European Chapter of the ACL*, pages 76–84, Gothenburg, Sweden. ACL.

Guilherme Lima, Alexandre Rademaker, and Rosario Uceda-Sosa. 2023. ULKB Logic: A HOL-based framework for reasoning over knowledge graphs. Under review.

Bill MacCartney and Christopher D Manning. 2007. Natural logic for textual inference. In *The ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 193–200.

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 *The Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. ELRA.

George A Miller. 1995. WordNet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Melanie Mitchell. 2023. How do we know how smart ai systems are? *Science*, 381(6654):adj5957.

Leonardo de Moura and Sebastian Ullrich. 2021. The lean 4 theorem prover and programming language. In *The 28th International Conference on Automated Deduction, Virtual Event, July 12–15, 2021, Proceedings 28*, pages 625–635. Springer.

Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. 2002. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Springer, Berlin.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. The University of Chicago Press and CSLI Publications, Chicago, IL and Stanford, CA.

Willard Van Quine. 1960. Carnap and logical truth. *Synthese*, 12:350–374.

Aarne Ranta. 1994. *Type-theoretical grammar*. Oxford University Press.

Stephan Schulz, Simon Cruanes, and Petar Vukmirović. 2019. Faster, higher, stronger: E 2.3. In *Automated Deduction – CADE 27*, pages 495–507. Springer.

The Coq Development Team. 2021. *The Coq Reference Manual: Release 8.14.0*.

Dag Westerståhl. 2019. Generalized Quantifiers. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, Winter 2019 edition. Metaphysics Research Lab, Stanford University.

Yoad Winter. 2016. *Elements of Formal Semantics: An Introduction to the Mathematical Theory of Meaning in Natural Language*. Edinburgh University Press.

A Quantitative Approach to Understand Self-Supervised Models as Cross-lingual Feature Extractors

Shuyue Stella Li[1⋆], **Beining Xu**[2⋆], **Xiangyu Zhang**[1⋆], **Hexin Liu**[3], **Wenhan Chao**[2], **Leibny Paola Garcia**[1]
[1]Center for Language and Speech Processing, Johns Hopkins University
[2]School of Computer Science and Engineering, Beihang University
[3]School of Electrical and Electronic Engineering, Nanyang Technological University
sli136, xzhan233, lgarci27@jhu.edu

Abstract

In this work, we study the features extracted by English self-supervised learning (SSL) models in cross-lingual contexts and propose a new metric to predict the quality of feature representations. Using automatic speech recognition (ASR) as a downstream task, we analyze the effect of model size, training objectives, and model architecture on the models' performance as a feature extractor for a set of topologically diverse corpora. We develop a novel metric, the Phonetic-Syntax Ratio (PSR), to measure the phonetic and synthetic information in the extracted representations using deep generalized canonical correlation analysis. Results show the contrastive loss in the wav2vec2.0 objective facilitates more effective cross-lingual feature extraction. There is a positive correlation between PSR scores and ASR performance, suggesting that phonetic information extracted by monolingual SSL models can be used for downstream tasks in cross-lingual settings. The proposed metric is an effective indicator of the quality of the representations and can be useful for model selection.[1]

1 Introduction

Self-Supervised Learning (SSL) has become a paradigm for learning feature representations from unlabeled data (Liu et al., 2023). In speech processing, self-supervised approaches for learning speech representation are often used to extract features for downstream tasks. These representations can replace the handcrafted feature such as Mel Spectrum or MFCC in many tasks as they are able to extract high-level properties in the speech data (Mohamed et al., 2022; Chung et al., 2019).

English SSL Models take advantage of the high availability of English data and outperform traditional feature extraction methods on a range of downstream tasks in English (Chen et al., 2022; Hsu et al., 2021; Liu et al., 2020). Since the acoustic and phonetic information of human speakers across languages share a level of similarity, it is crucial to study the cross-lingual transfer performance of English SSL models as a feature extractor for non-English audio data (Li et al., 2020; Cho et al., 2018). This will enhance our understanding of the composition of knowledge learned during pre-training, allowing more efficient use of data during model selection. Furthermore, if we are able to use English monolingual models effectively in multilingual downstream tasks, the high cost of training massive multilingual speech models such as XLSR (Babu et al., 2021; Conneau et al., 2021) and mSLAM (Bapna et al., 2022) can be reduced by explicitly incorporating architectural designs promoting cross-lingual transfer. Therefore, the first purpose of this paper is to investigate the factors that improve the ability of monolingual SSL models to extract useful speech representations for ASR tasks in typologically diverse languages.

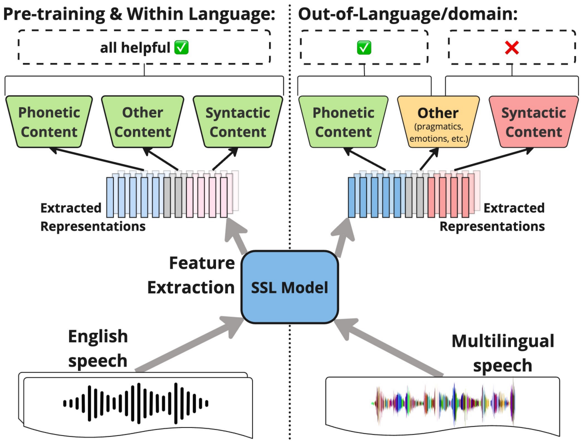


Figure 1: Speech data of English (in-domain) and other languages (out-of-domain) are passed through the SSL models to extract speech representations. All information is expected to aid downstream tasks in English while phonetic content is expected to be useful for out-of-domain downstream tasks; "other" content may include speaker information, etc.

[1] We make our work open-source for further explorations: https://github.com/stellali7/SSL_PSR

The second objective of our study is to analyze the amount of phonetic information versus syntactic information learned by the model during training, and how the phonetic-syntax composition in the model impacts the extracted features. Phonetic content directly impacts the learned phonological structure in the representations. Explicit integration of phonological knowledge has proven to be extremely successful in speech processing (Zhan et al., 2021). On the other hand, semantic and syntactic knowledge learning in the target language during fine-tuning is needed for ASR tasks so that the SSL models do not retain source language semantics and syntax, implying syntactic information might be harmful for cross-lingual feature extraction (Li et al., 2020).

As shown in Figure 1, we expect the pre-trained SSL models to efficiently extract phonetic, syntactic, and other contents to help downstream tasks in English (Chung et al., 2021). At the same time, the extracted phonetic information in out-of-domain and multilingual situations should also aid downstream performance. Therefore, we propose a novel metric to quantify the amount of helpful phonetic information. To the best of our knowledge, this study is the first to quantitatively understand the capabilities and limits of SSL models from a linguistic perspective. Our contributions include:

- We examine five SSL models with different sizes, data preparation methods, and training objectives by analyzing their cross-lingual generalizability as feature extractors on the ASR task.
- We propose a new metric, Phonology-Syntax Ratio (PSR), to measure the phonetic and syntactic content extracted by an SSL model on any given out-of-domain/language dataset. A higher PSR score correlates to a better ASR performance.
- We localize the phonetic content in the SSL model to specific layers using the trained layer-wise weights for the feature representations.

2 Related Work

2.1 Self-Supervised Models

Self-supervised learning (SSL) (Liu et al., 2023; Bengio et al., 2013; Raina et al., 2007) takes advantage of easily accessible unlabeled data to learn a model and then produces universal representations by solving upstream tasks (Liu et al., 2022b). Then, the pre-trained SSL model can be used to process unseen data based on its previous knowledge and handle multiple downstream tasks. SSL models have achieved superior performance in natural language processing (Devlin et al., 2019; Peters et al., 2018), computer vision (Chen et al., 2020; Misra and van der Maaten, 2020), speech processing (Chen et al., 2019; Chi et al., 2021), and especially ASR (Baevski and Mohamed, 2020; Ravanelli et al., 2020; Jiang et al., 2021). In our work, we study a number of SSL models and their feature extraction ability when presented with input from other languages.

2.2 Audio Feature Extraction

Before any downstream speech processing tasks, the audio data is converted to high-dimensional feature vectors through an audio feature extraction system (Moffat et al., 2015). Classic methods, such as Mel-Frequency Cepstral Coefficients (MFCCs), Linear Predictive Coding (LPC), and Perceptual Linear Prediction (PLP) extract cepstral coefficients that contain low-level acoustic features (Dave, 2013; Shanthi and Lingam, 2013). Researchers have also delved into neural-based models, leveraging pre-trained models on large-scale datasets to boost performance (Chi et al., 2021). While progress has been remarkable, challenges such as robustness to noise variations and interpretability of learned features continue to stimulate further research in this domain (Mohamed et al., 2022). In our work, we explore the robustness of the monolingual SSL models when generalized to multilingual settings, from which we interpret the features extracted by these models.

2.3 Automatic Speech Recognition (ASR)

ASR transcribes given audio to text in the script of the spoken language (Malik et al., 2021; Yu and Deng, 2016). Deep neural network (DNN) based techniques (Hinton et al., 2012) have boosted the accuracy of ASR by replacing the traditional Gaussian Mixture Model in cascaded systems involving separate acoustic, language, and lexicon components (Li et al., 2022). End-to-end models (Graves and Jaitly, 2014; Chorowski et al., 2014; Bahdanau et al., 2016; Collobert et al., 2016) have recently become a breakthrough in the speech community, directly translating an input speech sequence into an output text sequence with a single model. Some publicly available and commonly used toolkits include Kaldi (Povey et al., 2011), CMU Sphinx (Lee et al., 1990), SpeechBrain (Ravanelli et al., 2021) and ESPNet (Watanabe et al., 2018).

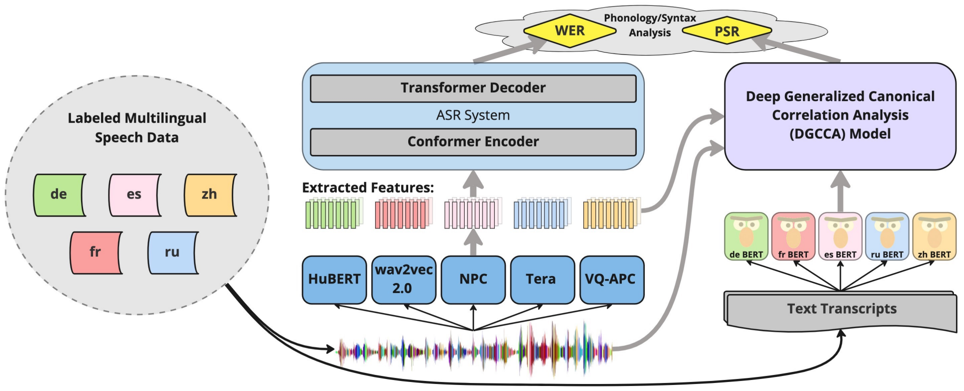


Figure 2: The pipeline to measure the performance of SSL model on different languages. We first use each SSL model as a feature extractor for data in each language and compute a WER score for the ASR task. Then, we calculate the PSR of the representations to analyze the correlation between the ASR performance and the PSR score.

2.4 Analysis Methods of SSL Models

There has been extensive research on analyzing supervised speech models (Belinkov and Glass, 2019; Palaskar et al., 2019; Prasad and Jyothi, 2020). However, research on SSL models, especially in the speech domain, is still relevantly under-explored. Some recent work in this field includes a similarity analysis of self-supervised speech representations, in which they only looked into simpler models such as APA, CPC, and MPC (Chung et al., 2021). Liu et al. (2022a) attempted to distinguish useful representations in SSL models for spoken language identification and reduce spurious information in the representations, but was limited to a specific task. Pasad et al. (2021) and Pasad et al. (2023) analyzed the layer-wise acoustic-linguistic content of pre-trained models by performing layer-independent Canonical Correlation Analysis (CCA) (Hardoon et al., 2004) on English data. However, since the features extracted by DNN models often have high dimensionality (Georgiou et al., 2020), CCA is limited in its ability to freely model complex nonlinear relationships.

2.5 Cross-lingual Knowledge Transfer

Cross-lingual transfer learning has gained attention in the field as it effectively mitigates resource constraints and language-specific challenges, but most importantly to our work, it requires the model to be able to adapt to unseen situations such as a new language (Khurana et al., 2023; Conneau et al., 2020). Effective cross-lingual transfer for speech processing requires the model to have a high-level understanding of both text linguistics and phonetics. Previous work has shown that multilingual models generalize well to target languages (Conneau et al., 2021; Singh et al., 2019; Radford et al., 2023). Lauscher et al. (2020) shows that the quality of the cross-lingual transfer is correlated with the linguistic similarity between the source and target languages. Inspired by this, we use English monolingual models in our work to better compare the linguistic distance between the pre-train data and the target data. Studying the generalization ability of monolingual models to unseen languages allows us to better analyze the learned representations and localize the factors that facilitate cross-lingual transfer for more efficient model design.

3 Analysis Methods

As shown in Figure 2, we first use the SSL models trained on English to extract speech representations on audio data from German (de), French (fr), Spanish (es), Russian (ru), and Chinese (zh). Then, we use the ASR task to evaluate the quality of the extracted features against a Mel Spectrum baseline in Section 3.1. We correlate the WER scores to traditional measures of linguistic distance in Section 3.2. Finally, we quantitatively evaluate the phonetic and syntactic content in the extracted features for each language, as described in Section 3.3.

3.1 Measuring Multilingual Generalizability

We use the standard ASR task on 5 genealogically and typographically diverse languages to evaluate the generalizability of the English SSL models as a cross-lingual feature extractor. To fairly compare the models, we freeze the parameters of the models and use the same downstream architecture (Conformer + Transformer) for all SSL models and the Mel Spectrum baseline feature extractor. We also use the same language model setup and beam size during decoding.

Our pipeline is shown in Figure 2. We select SSL models based on their training methods. These upstream SSL models can be categorized into **masked reconstruction model**: Tera (Liu et al., 2021b) and NPC (Liu et al., 2021a); **masked prediction model**: HuBERT (Hsu et al., 2021); **auto-regressive reconstruction model**: VQ-APC (Chung et al., 2020); and **contrastive model**: wav2vec2.0 (Baevski et al., 2020). Inspired by the setup in SUPERB (wen Yang et al., 2021) and ELMO (Peters et al., 2018), we take the weighted sum from all layers as the extracted representation, and the weight vector is updated during training.

For the downstream model, we use the Conformer (Gulati et al., 2020) as the encoder and the Transformer (Vaswani et al., 2017), which has achieved state-of-the-art (SOTA) results in many speech recognition tasks (Ma et al., 2021). During data analysis, we isolate the effect of the SSL model as a feature extractor by taking the difference (Δ) between the SSL feature extractor and the Mel Spectrum baseline performance. This eliminates any potential noise introduced by data size differences, speech formality levels, and other linguistic differences between languages, allowing a fair comparison between different SSL models. When decoding, we use a simple RNN as a language model and keep the parameters consistent across all tasks.

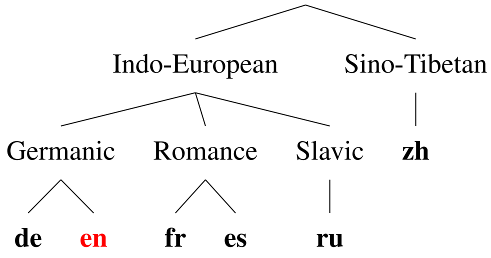


Figure 3: Phylogenetic Tree of Target Langauges

3.2 Measuring Linguistic Distance

We examine the performance of self-supervised models on languages across a diverse range of families and groups in order to investigate the relationship between model performance and linguistic distance. In our analysis, we employ the phylogenetic tree in Figure 3 derived from the theory of language evolution with genetic distance equaling the Levenshtein distance (Serva and Petroni, 2008) as a measure of linguistic distance. Since languages evolve with both their written and spoken forms, the phylogenetic tree will contain the most comprehensive information about the language.

3.3 Measuring Phonetic & Syntactic Content

In this section, we describe approaches to quantify phonetic and syntactic content in the extracted speech representations of SSL models.

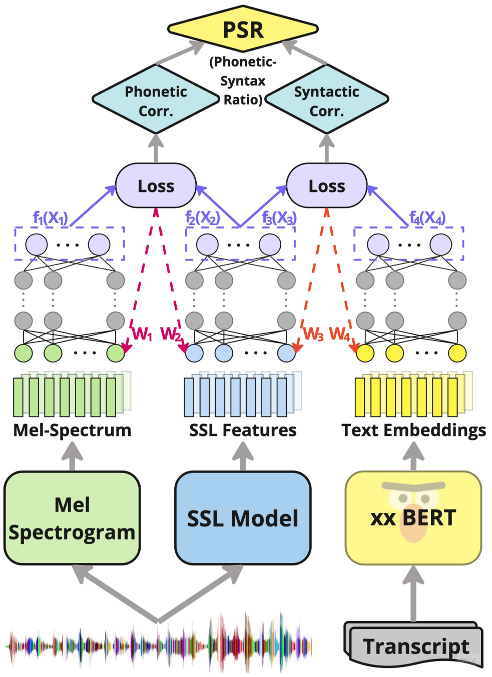


Figure 4: DGCCA pipeline. The model aims to compare the representation extracted by the SSL model to the pure acoustic representation (from Mel Spectrum) and pure syntactic/semantic representation (from BERT).

3.3.1 DGCCA

In order to better analyze the phonetic and syntactic content of the features, we use a tool called Deep Generalized Canonical Correlation Analysis (DGCCA), which is a deep learning technique that measures the nonlinear relationship between arbitrarily many views of the data and learns a view-independent representation (Benton et al., 2019). DGCCA effectively quantifies the phonetic and syntactic content of SSL models when treating the features extracted with different models as different views of the same data.

As shown in Figure 4, DGCCA takes N pairs of data vectors across J views as input and returns a correlation score as a measure of the similarity between the vectors. Using standard backpropagation to optimize the weight matrices $W_j = \{W_1^j, \ldots, W_{K_j}^j\}$, we try to find the linear transformation $U_j \in \mathbb{R}^{d_j \times N}$ of $f_i(X_j) \in \mathbb{R}^{o_j}$ constrained by $GG^T = I_r$ such that:

$$\underset{U_j \in \mathbb{R}^{d_j \times N}, G \in \mathbb{R}^{r \times N}}{\text{minimize}} \sum_{j=1}^{J} \|G - U_j^T f_j(X_j)\|_F^2 , \quad (1)$$

where $X_j \in \mathbb{R}^{d_j \times N}$ is the input feature vectors of the j^{th} view; f_j is the function learned using a multilayer perceptron of K_j layers; d_j is the dimension of the j^{th} view and r is the dimension of the learned representation G.

In our case, N is the number of utterances in the test data, where we have the SSL features and Mel Spectrum features of each utterance, as well as the BERT representations of its transcript. The monolingual BERT model in each target language is used when extracting the textual representations.

Features extracted by the SSL models, pure phonetic features (Mel Spectrum), and pure textual features (BERT representations) can be considered as different views (f_i) of the data. The correlation scores between different views are the loss of the converged DGCCA network. We compute the correlation scores between each of the latter two views and the SSL features. The correlation scores between the SSL features and the Mel Spectrum measure the **Phonetic Content** in the extracted features; the correlation scores between the SSL features and the BERT representations measure the **Syntactic Content** in the extracted features.

3.3.2 Phonetic-Syntax Ratio (PSR)

We introduce a new metric: the Phonetic-Syntax Ratio (PSR) in order to quantitatively investigate the phonetic and syntactic content on SSL representation. As described in Section 3.3.1, the similarity to phonetic features and the similarity to syntactic features of the SSL representations are both optimized and quantified as correlation scores when training the DGCCA network. We define the PSR as the ratio between the phonetic correlation score and syntactic correlation score, weighted equally among all data points:

$$PSR = (\frac{1}{n}\sum_{i=1}^{n} \frac{phonetic\ score_i}{syntax\ score_i} - 1) \cdot 100\%, \quad (2)$$

where phonetic scores and syntax scores are the output of DGCCA when the SSL representations are fed in with the Mel Spectrum and BERT contextualized embedding, respectively. The PSR score is model-agnostic and language-agnostic, and can be used for a range of contrastive analysis for inferring cross-lingual transferability.

4 Experimental Setup

4.1 Datasets

We investigate the cross-lingual adaptation capability of English SSL models in five languages. For training the ASR models, we use the Mozilla Common Voice 5.1 dataset (Ardila et al., 2020) for German, French, Spanish, and Russian, and we use the OpenSLR ST-CMDS-20170001_1 Free ST Chinese Mandarin Corpus[2] for Chinese. The Common Voice English test set is used for DGCCA analysis. More details about the datasets are in Table 1.

Lang	hr	voices	train	dev	test
de	751	11,731	196,464	15,341	15,341
fr	605	11,960	254,863	15,621	15,621
es	522	18,906	138,878	14,860	14,860
ru	117	927	13,189	7,242	7,307
zh	-	-	92,280	4,299	4,483
en	1933	61528	435,947	16,029	16,029

Table 1: Dataset description; the number of hours, voices, and utterances for each split. Hour and voice statistics for the Chinese corpus are not available as it is distributed after preprocessing. The number of speakers for the Chinese dataset is 855. Train and dev splits of English were not used.

4.2 Multilingual Generalizability Setup

We use the ASR performance on a range of typologically diverse languages as a metric to infer the models' multilingual generalizability. In order to fairly compare the performance of each SSL model in different language datasets, we use the same downstream model for all languages and features and focus on the within-language difference between the SSL model and the baseline model.

Self-supervised feature extractors We examine a number of English SSL speech models including HuBERT (Hsu et al., 2021), wav2vec 2.0 (Collobert et al., 2016), NPC (Liu et al., 2021a), TERA (Liu et al., 2021b), and VQ-APC (Chung et al., 2020) with model details shown in Table 2. Unlike the baseline model, we use a smaller learning rate considering that self-supervised training usually uses a small learning rate. We use a learning rate of 0.0025 with 40000 warmup steps.

Model architectures After multilingual features are extracted, we use a standard Conformer encoder and a Transformer decoder in our downstream ASR model and a stacked RNN as the language model

[2] http://www.openslr.org/38

Model	architecture	train objective	model size	pre-train	input	stride
HuBERT-Base HuBERT-Large	CNN + Transformer	Predictive	95m 317m	LS-960 LL-60k	wav	20ms
wav2vec2-Base wav2vec2-Large	CNN + Transformer	Contrastive + Diversity	95m 317m	LS-960 LV-53.2k	wav	20ms
NPC	Masked Conv Block	L1 Reconstruction	19.4m	LS-C-360	Mel	10ms
TERA-Base	Unidirectional LSTM + Prediction Network	L1 Reconstruction	21.3m	LS-C-100	Mel	10ms
VQ-APC	Unidirectional LSTM	L1 Reconstruction	4.63m	LS-C-360	Mel	10ms

Table 2: SSL Model Summary. For the pre-training data description, LS = Librispeech, LS-C = Librispeech-clean, LL = Libri-light, and LV = Libri-vox.

during decoding. More details on the ASR model architecture and training are in Appendix A.

4.3 PSR Computation

We use the DGCCA pipeline shown in Figure 4 to compute the PSR scores for each langauge. The DGCCA model used consists of an MLP network with a Linear layer, a Sigmoid function, and a Batch Norm layer. Each group of tensors has one MLP network, and its output is passed into the DGCCA loss. We used SGD to optimize the network with a learning rate of 1e-6. We use features extracted by the HuBERT model from five different languages (German, French, Spanish, Russian, and English) and also extract its corresponding Mel Spectrum and BERT features. Chinese PSR is not reported because CER was used to evaluate the ASR performance, hence the comparison across languages would not be fair (more details in Section 5.2). When calculating the correlation scores, we use the test set in each target language as input to the DGCCA model with a batch size of 32. Details on the implementation and hardware of the SSL models and the DGCCA model can be found in Appendix B.

5 Results and Analysis

5.1 Multilingual Generalizability

Results from the multilingual ASR tasks are shown in Table 3, with both WER scores and the difference from the Mel Spectrum baseline (Δ).

In the zero-shot setting, it is generally expected that the SSL feature extractor trained on English, without any domain adaptation, performs poorly on the cross-lingual ASR tasks compared to the Mel spectrum baseline. Although it can extract higher-dimensional features, additional English syntactic information in the SSL model can be projected onto the new language (Georgiou et al., 2020). Therefore, the purpose of this experiment is not to improve the SOTA results but rather to probe the SSL models for further phonetic-syntactic analysis.

There are five SSL models being evaluated in this experiment in five languages. The column Avg on the right marginal of Table 3 shows the overall performance of each SSL model in all languages. In general, wav2vec2.0-Large significantly outperforms other feature extractors and has a consistent result across languages. There are two instances in which wav2vec2.0-Large outperforms the pure acoustic Mel Spectrum baseline. This can be attributed to the cross-lingual phonetic information transfer that the model learned from English pre-training.

5.1.1 Effect of Training Objectives

The HuBERT and wav2vec2.0 models consistently perform better than NPC, TERA, and VQ-APC. HuBERT and wav2vec2.0 both effectively combine CNN encoders with Transformers in their architecture. The attention mechanism allows the models to effectively encode speech features into the latent embedding space and learn contextualized representations. Both HuBERT and wav2vec2.0 use similar architectures and identical pre-training data and setups. However, HuBERT as a cross-lingual feature extractor does not perform as well due to its predictive loss compared to the **contrastive loss** of wav2vec2.0. The masked prediction task during HuBERT pre-training forces the model to learn the language model as well as the acoustic model from continuous English speech inputs (Hsu et al., 2021), so the model might be overfitted to English syntax.

Now we discuss the performance of NPC, TERA, and VQ-APC, which are significantly smaller than wav2vec2.0 and HuBERT both in model and data size. TERA and NPC have comparable model sizes, training objectives, input format, and stride during pre-training, but TERA outperforms NPC with less than one-third of the training data. This is

Model/Lang	de	Δ	fr	Δ	es	Δ	ru	Δ	zh	Δ	Avg.	Δ
Mel (Baseline)	10.0	-	**15.8**	-	**11.5**	-	7.9	-	9.4	-	**10.92**	-
HuBERT-BASE	11.3	1.3	16.5	**0.7**	13.1	1.6	7.8	-0.1	9.8	0.4	11.70	0.78
HuBERT-LARGE	12.4	2.4	16.6	0.8	12.0	**0.5**	8.3	0.4	**9.1**	**-0.3**	11.68	0.76
wav2vec2-BASE	11.8	1.8	16.7	0.9	13.4	1.9	8.5	0.6	9.8	0.4	12.04	1.12
wav2vec2-LARGE	**9.2**	**-0.8**	16.6	0.8	12.3	0.8	**7.6**	**-0.3**	9.4	0	11.04	**0.10**
NPC	16.2	6.2	18.1	2.3	16.1	4.6	11.0	3.1	10.7	1.3	14.42	3.5
TERA-BASE	15.6	5.6	17.1	1.3	14.8	3.3	10.3	2.4	10.0	0.6	13.56	2.64
VQ-APC	13.5	3.5	17.2	1.4	17.3	5.8	12.1	4.2	10.8	1.4	14.18	3.26
Avg.	12.86	2.86	16.97	1.17	14.14	2.64	9.37	1.47	9.94	0.54	-	-

Table 3: Word Error Rate (WER) of German (de), French (fr), Spanish (es), and Russian (ru). For Chinese (zh), we apply Character Error Rate (CER) as the evaluation metric. Δ is the difference from Baseline, the lower the better. wav2vec2.0-LARGE achieves the best performance and the Transformer-based models generally perform better.

due to the alterations in the time, frequency, and magnitude axes of the data during pre-training, which increases **data diversity** and enforces accurate phoneme prediction (Liu et al., 2021b). On the other hand, VQ-APC achieves comparable results as NPC with a much smaller model size. With all the other setups identical, this suggests that the **sequential structure** learned by the Unidirectional LSTM (APC) and the **quantization layers** are more effective at capturing speech representations than convolutional blocks in NPC, implying that speech should be treated as sequential data.

5.1.2 Effect of Model Size

Comparing the HuBERT-BASE / HuBERT-LARGE and wav2vec2.0-BASE / wav2vec2.0-LARGE pairs gives insight into the effect of model size on downstream ASR tasks. The LARGE models generally perform better than the BASE models. This is consistent with a previous study by Pu et al. (2021), in which they empirically showed that scaling SSL models results in improvements in both L1 loss and accuracy on downstream tasks consistent with the power law. Larger models are also more data-efficient when labeled data is scarce. The advantage of the LARGE model over the BASE model is especially apparent on the wav2vec2.0 pair, as wav2vec2.0-LARGE consistently performs better across all languages. As discussed in Section 5.1.1, the more efficient use of data in HuBERT-LARGE may have caused it to learn even more syntactic and semantic representation, which does not benefit cross-lingual speech feature extraction.

5.2 Linguistic Analysis

Now we discuss the performances of all five languages based on their average scores. Smaller Δ indicates better generalizability. According to the phylogenetic tree shown in Figure 3, both German and English belong to the Germanic branch; French, Spanish, and Russian are in different language groups as English; Chinese belongs to another language family. As shown in Table 3, English SSL models have better generalizability in French than in German. This is because French has a profound phonological influence on the development of English (Roth, 2010), and the latter not only borrows some French pronunciation rules, but also shares contextual phonetic similarities of pitch contours (So and Best, 2014). For German, although it appears to have poor SSL performance with high Δ values, the absolute WER is the lowest among German, French, and Spanish, which have similar training sizes. From this, it can be observed that SSL representations has diminishing returns in high-resource situations.

Features extracted by the SSL models also perform well in Russian and Chinese ASR tasks. This might seem surprising, but it is because both Russian and Chinese are low-resource with less than 100k utterances. This demonstrates the robustness of SSL models in low-resource settings and establishes promising directions to generalize to other low-resource languages. Moreover, although Chinese is in the Sino-Tibetan language family, it actually has some phonotactic similarities with English (Ann Burchfield and Bradlow, 2014; Yang, 2021). It is important to note that the CER was used as the metric for Chinese ASR to avoid additional noise introduced by a word segmentation model, so the Chinese results should only be compared across models rather than across languages.

Analysis by linguistic distance can provide some plausible explanations for the results, but there still exist some inconsistencies. These inconsistencies motivate our next section, PSR Analysis, in which we use our novel metric to explain the model performance by categorizing and quantifying linguistic information in the extracted representations.

5.3 PSR Analysis

PSR scores of HuBERT-BASE on English and the target languages are shown in Table 4. As described in Equation 2, the larger the PSR, the more phonetic content in the feature set. First, to validate the PSR scale, we test the SSL features extracted from an English corpus by the SSL model. The PSR value from the English corpus is close to zero, which conforms with the intuition that the English-trained HuBERT model is able to extract useful information in both the phonetic and syntactic fields.

Lang	en	de	fr	es	ru
PSR	.01	.15	.16	.13	.23
WER Δ	-	1.3	0.7	1.6	-0.1

Table 4: PSR Results for Target Languages. A positive PSR means that the phonetic content in the extracted representations is stronger than the syntactic content.

Combined with the information in Table 3, we show that there is a positive correlation between the PSR scores of the feature group and the ASR performance of the model in that language. For example, the Δ value of HuBERT-BASE on German is higher (worse) than that of French and lower (better) than that of Spanish as shown in Table 3, and we see the corresponding relationship of their PSR values in Table 4: German PSR is lower (worse, less phonetic info) than French and higher (better, more phonetic info) than Spanish. This phenomenon indicates that the more phonetic information contained in a set of features, the better the performance of that set of features on cross-lingual or out-of-domain downstream tasks. Therefore, when the SSL model trained with English models is applied to the non-English corpus, *phonetic features are the main contributors to effective information compared with syntactic features*.

5.4 Layer Weights Analysis

All PSR scores shown in Table 4 are positive, suggesting that the features extracted by speech SSL models tend to have more phonetic information than syntactic information. This is partially due to the fact that the weighted sum of layers is used as input features to the ASR model and that the weights are optimized during training to put more emphasis on the phonetic information. Figure 5 shows the magnitude of the weights across all layers of HuBERT-BASE.

First, the layer-wise trend is consistent across all languages, suggesting each layer contains similar information even when trained on different datasets, i.e., the weights get updated similarly given the same task. The optimized weights gravitate toward layers that are crucial for the ASR task. The positive correlation between the ASR and PSR scores implies that the layers with large weights contribute to the high PSR scores, i.e. have denser phonetic than syntactic information. From Figure 5, Layers 4, 11, and 12 contribute significantly to the extracted features. Since lower layers contain lower-level information and vise versa, Layer 4 (and its adjacent layers) contain low or intermediate-level information on acoustic and phonetics important for the ASR task. The last two layers are the most salient because they contain high-level information related to human phonetics. Additionally, the weight for Layer 4 is larger in German and French, which are closer to English. This shows that when the pre-training and target languages are highly similar, the low-level phonetic features become more helpful. Our work to localize the phonetic content encoded in specific layers of HuBERT draws similar conclusions with Pasad et al. (2021) and Pasad et al. (2023), which localized various acoustic and linguistic properties in SSL models using CCA.

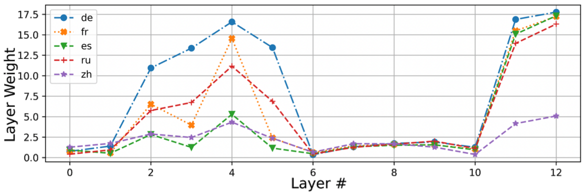


Figure 5: Layer-wise Weight Analysis.

6 Conclusion

In this work, we studied English self-supervised speech models and probed for the phonetic and syntactic content in the extracted speech representations. We accomplished this using the SSL models as a feature extractor for downstream ASR task in multiple languages. Higher multilingual adaptability of a model is found to be positively correlated to the amount of phonetic information in the extracted representations. Most importantly, we propose a novel metric - the Phonetic-Syntax Ratio (PSR) - to quantify the phonetic and syntactic composition in the representations. PSR can serve as an effective indicator during model selection. We were also able to localize the phonetic information to certain layers in the SSL model. This is a call to other researchers to design smarter objectives when pre-training large models (such as focusing more on phonetic information learning) rather than simply increasing the model size.

Limitations

There are several limitations to our work. First, the value of our PSR was only tested on HuBERT due to limited computing resources. Although the scores reflect the ratio of acoustic and linguistic information in the features extracted by the SSL model, the performance of the corresponding downstream ASR task is not yet empirically shown in every SSL model. Second, the parameters in the SSL models are frozen during ASR training. Multilingual adaptability might be evaluated differently by unfreezing some or all layers of the SSL feature extractor. Finally, we did not calculate the PSR value for Chinese, as we did not find it to be a valuable data point given the Chinese ASR results are reported in CER only. Our choice to evaluate English SSL models is motivated by the abundance or English data, but other monolingual or multilingual models could be used given the abundance of data in the chosen langauge(s). For future directions, we believe that exploring spurious correlations among language pairs (e.g. phonotactical similarities between Chinese and English) is a fruitful direction that might shed light on language selection during cross-lingual transfer in speech models.

References

L. Ann Burchfield and Ann R. Bradlow. 2014. Syllabic reduction in Mandarin and English speech. *The Journal of the Acoustical Society of America*, 135(6):EL270–EL276.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France. European Language Resources Association.

Arun Babu, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, et al. 2021. Xls-r: Self-supervised cross-lingual speech representation learning at scale. *arXiv preprint arXiv:2111.09296*.

Alexei Baevski and Abdelrahman Mohamed. 2020. Effectiveness of self-supervised pre-training for asr. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7694–7698.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc.

Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philémon Brakel, and Yoshua Bengio. 2016. End-to-end attention-based large vocabulary speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4945–4949.

Ankur Bapna, Colin Cherry, Yu Zhang, Ye Jia, Melvin Johnson, Yong Cheng, Simran Khanuja, Jason Riesa, and Alexis Conneau. 2022. mslam: Massively multilingual joint pre-training for speech and text. *arXiv preprint arXiv:2202.01374*.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828.

Adrian Benton, Huda Khayrallah, Biman Gujral, Dee Ann Reisinger, Sheng Zhang, and Raman Arora. 2019. Deep generalized canonical correlation analysis. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 1–6, Florence, Italy. Association for Computational Linguistics.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, and Furu Wei. 2022. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Yi-Chen Chen, Sung-Feng Huang, Hung-yi Lee, Yu-Hsuan Wang, and Chia-Hao Shen. 2019. Audio word2vec: Sequence-to-sequence autoencoding for unsupervised learning of audio segmentation and representation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(9):1481–1493.

Po-Han Chi, Pei-Hung Chung, Tsung-Han Wu, Chun-Cheng Hsieh, Yen-Hao Chen, Shang-Wen Li, and Hung-yi Lee. 2021. Audio albert: A lite bert for self-supervised learning of audio representation. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 344–350.

Jaejin Cho, Murali Karthick Baskar, Ruizhi Li, Matthew Wiesner, Sri Harish Mallidi, Nelson Yalta, Martin Karafiát, Shinji Watanabe, and Takaaki Hori. 2018. Multilingual sequence-to-sequence speech recognition: Architecture, transfer learning, and language modeling. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 521–527.

Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. End-to-end continuous speech recognition using attention-based recurrent nn: First results. *arXiv preprint arXiv:1412.1602*.

Yu-An Chung, Yonatan Belinkov, and James Glass. 2021. Similarity analysis of self-supervised speech representations. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3040–3044.

Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass. 2019. An Unsupervised Autoregressive Model for Speech Representation Learning. In *Proc. Interspeech 2019*, pages 146–150.

Yu-An Chung, Hao Tang, and James Glass. 2020. Vector-Quantized Autoregressive Predictive Coding. In *Proc. Interspeech 2020*, pages 3760–3764.

Ronan Collobert, Christian Puhrsch, and Gabriel Synnaeve. 2016. Wav2letter: an end-to-end convnet-based speech recognition system. *arXiv preprint arXiv:1609.03193*.

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. 2021. Unsupervised Cross-Lingual Representation Learning for Speech Recognition. In *Proc. Interspeech 2021*, pages 2426–2430.

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.

Namrata Dave. 2013. Feature extraction methods lpc, plp and mfcc in speech recognition. *International journal for advance research in engineering and technology*, 1(6):1–4.

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.

Theodoros Georgiou, Yu Liu, Wei Chen, and Michael Lew. 2020. A survey of traditional and deep learning-based feature descriptors for high dimensional data in computer vision. *International Journal of Multimedia Information Retrieval*, 9(3):135–170.

Alex Graves and Navdeep Jaitly. 2014. Towards end-to-end speech recognition with recurrent neural networks. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1764–1772, Bejing, China. PMLR.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented Transformer for Speech Recognition. In *Proc. Interspeech 2020*, pages 5036–5040.

David R. Hardoon, Sandor Szedmak, and John Shawe-Taylor. 2004. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664.

Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. 29:3451–3460.

Dongwei Jiang, Wubo Li, Ruixiong Zhang, Miao Cao, Ne Luo, Yang Han, Wei Zou, Kun Han, and Xiangang Li. 2021. A further study of unsupervised pretraining for transformer based speech recognition. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6538–6542.

Sameer Khurana, Nauman Dawalatabad, Antoine Laurent, Luis Vicente, Pablo Gimeno, Victoria Mingote, and James Glass. 2023. Improved cross-lingual transfer learning for automatic speech translation. *arXiv preprint arXiv:2306.00789*.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

K.-F. Lee, H.-W. Hon, and R. Reddy. 1990. An overview of the sphinx speech recognition system. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(1):35–45.

Jinyu Li et al. 2022. Recent advances in end-to-end automatic speech recognition. *APSIPA Transactions on Signal and Information Processing*, 11(1).

Song Li, Lin Li, Qingyang Hong, and Lingling Liu. 2020. Improving Transformer-Based Speech Recognition with Unsupervised Pre-Training and Multi-Task Semantic Knowledge Learning. In *Proc. Interspeech 2020*, pages 5006–5010.

Alexander H. Liu, Yu-An Chung, and James Glass. 2021a. Non-Autoregressive Predictive Coding for Learning Speech Representations from Local Dependencies. In *Proc. Interspeech 2021*, pages 3730–3734.

Andy T. Liu, Shang-Wen Li, and Hung-yi Lee. 2021b. Tera: Self-supervised learning of transformer encoder representation for speech. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2351–2366.

Andy T. Liu, Shu-wen Yang, Po-Han Chi, Po-chun Hsu, and Hung-yi Lee. 2020. Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6419–6423.

Hexin Liu, Leibny Paola Garcia Perera, Andy W. H. Khong, Eng Siong Chng, Suzy J. Styles, and Sanjeev Khudanpur. 2022a. Efficient self-supervised learning representations for spoken language identification. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1296–1307.

Shuo Liu, Adria Mallol-Ragolta, Emilia Parada-Cabaleiro, Kun Qian, Xin Jing, Alexander Kathan, Bin Hu, and Björn W. Schuller. 2022b. Audio self-supervised learning: A survey. *Patterns*, 3(12):100616.

Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. 2023. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):857–876.

Pingchuan Ma, Stavros Petridis, and Maja Pantic. 2021. End-to-end audio-visual speech recognition with conformers. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7613–7617.

Mishaim Malik, Muhammad Kamran Malik, Khawar Mehmood, and Imran Makhdoom. 2021. Automatic speech recognition: a survey. *Multimedia Tools and Applications*, 80(6):9411–9457.

Ishan Misra and Laurens van der Maaten. 2020. Self-supervised learning of pretext-invariant representations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6706–6716.

David Moffat, David Ronan, and Joshua D Reiss. 2015. An evaluation of audio feature extraction toolboxes. *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx-15)*, pages 277–283.

Abdelrahman Mohamed, Hung-yi Lee, Lasse Borgholt, Jakob D. Havtorn, Joakim Edin, Christian Igel, Katrin Kirchhoff, Shang-Wen Li, Karen Livescu, Lars Maaløe, Tara N. Sainath, and Shinji Watanabe. 2022. Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1179–1210.

Shruti Palaskar, Vikas Raunak, and Florian Metze. 2019. Learned in speech recognition: Contextual acoustic word embeddings. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6530–6534.

Ankita Pasad, Ju-Chieh Chou, and Karen Livescu. 2021. Layer-wise analysis of a self-supervised speech representation model. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 914–921.

Ankita Pasad, Bowen Shi, and Karen Livescu. 2023. Comparative layer-wise analysis of self-supervised speech models. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

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.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The kaldi speech recognition toolkit. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, CONF. IEEE Signal Processing Society.

Archiki Prasad and Preethi Jyothi. 2020. How accents confound: Probing for accent information in end-to-end speech recognition systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3739–3753, Online. Association for Computational Linguistics.

Jie Pu, Yuguang Yang, Ruirui Li, Oguz Elibol, and Jasha Droppo. 2021. Scaling Effect of Self-Supervised Speech Models. In *Proc. Interspeech 2021*, pages 1084–1088.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine Mcleavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *Proceedings of the 40th International*

Conference on Machine Learning, volume 202 of *Proceedings of Machine Learning Research*, pages 28492–28518. PMLR.

Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. 2007. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 759–766, New York, NY, USA. Association for Computing Machinery.

Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, et al. 2021. Speechbrain: A general-purpose speech toolkit. *arXiv preprint arXiv:2106.04624*.

Mirco Ravanelli, Jianyuan Zhong, Santiago Pascual, Pawel Swietojanski, Joao Monteiro, Jan Trmal, and Yoshua Bengio. 2020. Multi-task self-supervised learning for robust speech recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6989–6993.

Isabel Roth. 2010. Explore the influence of french on english. *Leading Undergraduate Work in English Studies*, 3:255–261.

M. Serva and F. Petroni. 2008. Indo-european languages tree by levenshtein distance. *Europhysics Letters*, 81(6):68005.

Therese S Shanthi and Chelpa Lingam. 2013. Review of feature extraction techniques in automatic speech recognition. *International Journal of Scientific Engineering and Technology*, 2(6):479–484.

Jasdeep Singh, Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2019. Xlda: Cross-lingual data augmentation for natural language inference and question answering. *arXiv preprint arXiv:1905.11471*.

Connie K So and Catherine T Best. 2014. Phonetic influences on english and french listeners'assimilation of mandarin tones to native prosodic categories. *Studies in Second Language Acquisition*, 36(2):195–221.

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.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. ESPnet: End-to-End Speech Processing Toolkit. In *Proc. Interspeech 2018*, pages 2207–2211.

Shu wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y. Lin, Andy T. Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, Tzu-Hsien Huang, Wei-Cheng Tseng, Ko tik Lee, Da-Rong Liu, Zili Huang, Shuyan Dong, Shang-Wen Li, Shinji Watanabe, Abdelrahman Mohamed, and Hung yi Lee. 2021. SUPERB: Speech Processing Universal PERformance Benchmark. In *Proc. Interspeech 2021*, pages 1194–1198.

Jing Yang. 2021. Comparison of vots in mandarin–english bilingual children and corresponding monolingual children and adults. *Second Language Research*, 37(1):3–26.

Dong Yu and Li Deng. 2016. *Automatic speech recognition*, volume 1. Springer.

Qingran Zhan, Xiang Xie, Chenguang Hu, and Haobo Cheng. 2021. A self-supervised model for language identification integrating phonological knowledge. *Electronics*, 10(18):2259.

A ASR Model Architecture and Training

The downstream ASR model is composed of a Conformer encoder and a Transformer decoder. The encoder consists of 12 blocks and 4 attention heads with an output size of 256, and the decoder consists of 6 blocks. We use an Adam optimizer with 25000 warmup steps. The model is initialized with Xavier Uniform distribution and trained for 50 epochs with early stopping. We take the average of the best 10 models as the prediction model in the ASR task. To focus on the performance of the SSL feature extractor, we used a simple stacked RNN as the language model during decoding. The RNN language model has 2 layers and each layer has 650 units optimized by the SGD algorithm. We train this language model for 20 epochs and only keep the best one as our language model. During decoding, we use 0.3 as the weight of the language model and decode data with a beam size of 10.

B Implementation and Hardware

We obtain the upstream SSL models and DGCCA model from the S3PRL Speech Toolkit (wen Yang et al., 2021). The ASR training and DGCCA computation were both done on NVIDIA Tesla V100 for all model-language pairs. The average time of each experiment depends on the dataset size but cost about one week to complete on two GPUs for ASR and one day for DGCCA.

DEF2VEC: Extensible Word Embeddings from Dictionary Definitions

Irene Morazzoni, **Vincenzo Scotti** and **Roberto Tedesco**
DEIB, Politecnico di Milano
Via Golgi 42, 20133, Milano (MI), Italy
irene.morazzoni@mail.polimi.it vincenzo.scotti@polimi.it
roberto.tedesco@polimi.it

Abstract

DEF2VEC introduces a novel paradigm for word embeddings, leveraging dictionary definitions to learn semantic representations. By constructing term-document matrices from definitions and applying *Latent Semantic Analysis* (LSA), DEF2VEC generates embeddings that offer both strong performance and extensibility. In evaluations encompassing *Part-of-Speech tagging*, *Named Entity Recognition*, *chunking*, and *semantic similarity*, DEF2VEC often matches or surpasses state-of-the-art models like WORD2VEC, GLOVE, and FASTTEXT. Our model's second factorised matrix resulting from LSA enables efficient embedding extension for out-of-vocabulary words. By effectively reconciling the advantages of dictionary definitions with LSA-based embeddings, DEF2VEC yields informative semantic representations, especially considering its reduced data requirements. This paper advances the understanding of word embedding generation by incorporating structured lexical information and efficient embedding extension.

1 Introduction

Nowadays, *semantic representations* are the core of Natural Language Processing (NLP), allowing machines to capture the intricate relationships between words and their meanings (Liu et al., 2020b). *Word embeddings* have emerged as a cornerstone of this representation, enabling the translation of textual data into numerical vectors that encapsulate semantic nuances (Jurafsky and Martin, 2023, Chapter 6). These embeddings facilitate a wide range of NLP tasks, from sentiment analysis to machine translation, by endowing algorithms with means to comprehend and manipulate language, (Raffel et al., 2020; Brown et al., 2020; Sanh et al., 2022).

Contemporary advances in NLP have witnessed a paradigm shift from traditional *static word embeddings* to *dynamic contextual embeddings*, enabled by *Transformer* network-based models (Vaswani et al., 2017). Contextual embeddings, such as those derived from BERT (Devlin et al., 2019), GPT (Radford et al., 2018, 2019; Brown et al., 2020; OpenAI, 2023), and their variants, capture not just the inherent meaning of a word, but also its significance within the surrounding context (Liu et al., 2020a; Wang et al., 2022). While these contextual embeddings have undeniably revolutionised NLP benchmarks, the utilisation of static word embeddings persists. These embeddings, often generated through methods like WORD2VEC (Mikolov et al., 2013a,b), GLOVE (Pennington et al., 2014), and FASTTEXT (Bojanowski et al., 2017), remain valuable for their simplicity, interpretability, and efficiency (Hosseini et al., 2023).

In this paper, we introduce DEF2VEC, an innovative approach to constructing word embeddings that marries traditional static embeddings' virtues with dictionary definitions' information-rich nature. Recognising the enduring utility of static embeddings alongside the dominance of contextual embeddings, we propose exploiting the structured knowledge encapsulated in dictionary definitions. This unique strategy involves building term-document matrices from the definitions of words, which are subsequently factorised using *Latent Semantic Analysis* (LSA) (Deerwester et al., 1989, 1990). Remarkably, the embeddings extracted from this factorisation exhibit competitive performance on various tasks, often matching or even outperforming state-of-the-art static embeddings.

Our primary contribution lies in the extensibility of DEF2VEC embeddings. With a twofold factorisation approach, DEF2VEC generates robust embeddings from existing definitions and offers a seamless mechanism for accommodating new words into the embedding space. This scalability addresses a longstanding challenge in word embeddings, where adding new words necessitates retraining the entire model. The presented empirical evaluations across multiple tasks underscore the strengths of DEF2VEC embeddings, making it a valuable alternative for static embedding models.

We divide this paper into the following sections. In Section 2, we discuss related works in the domain of word embeddings. Section 3 elaborates on the DEF2VEC model, detailing the term-document matrix construction and the factorisation process. Section 4 presents the data set employed for experimentation. Section 5 outlines our evaluation methodology, encompassing benchmarks and metrics. Subsequently, Section 6 dissects these results, offering insights into the efficacy of DEF2VEC embeddings. Finally, Section 7 summarises our findings and proposes possible future extensions.

2 Related works

Nowadays, word and sentence embeddings are fundamental tools in NLP, capturing the essence of language in numerical representations. In this section, we provide a taxonomy of embedding models, classifying them based on the level of the linguistic unit (*words vs. sentences/documents*) and the nature of the representation (*static vs. contextual* for word embeddings, *parametric vs. non-parametric* for sentence/document embeddings).

2.1 Word Embeddings

Word embeddings are the cornerstone of NLP, encapsulating word meanings in vector spaces. These embeddings can be broadly categorised into two main classes: static and contextual.

Static (or *shallow*) word embeddings capture word meanings independently of context, representing words as fixed vectors. Examples of this category include WORD2VEC (Mikolov et al., 2013a,b), GLOVE (Pennington et al., 2014), and FASTTEXT (Bojanowski et al., 2017), which generate embeddings through methods like *skip-gram*, *Continuos-Bag-of-Words* (CBoW), global *co-occurrence statistics*, and *sub-word information*.

Contextual (or deep) embeddings, on the other hand, integrate *contextual information* to produce dynamic representations. Models like ELMO (Peters et al., 2018), BERT (Devlin et al., 2019), GPT (Radford et al., 2018), and their variants generate embeddings by considering the surrounding words or sentences (i.e., the context), resulting in nuanced and context-sensitive representations (Liu et al., 2020b). These models build on top of word embeddings, starting from static representations and using *Deep Neural Networks* (DNNs) for sequence processing such as *Recurrent* neural network (Elman, 1990; Hochreiter and Schmidhuber, 1997) or *Transformer* (Vaswani et al., 2017) neural networks.

2.2 Sentence and Document Embeddings

While word embeddings capture individual word meanings, sentence and document embeddings aim to capture the meaning of larger textual units. These embeddings can be classified into parametric and non-parametric based on their approach to representation and generation.

Parametric sentence embeddings are generated using neural network architectures trained to produce fixed-size vectors from input sequences (e.g., sentences). SKIP-THOUGHT vectors (Kiros et al., 2015), SENT2VEC (Pagliardini et al., 2018), and SENTENCE-BERT (Reimers and Gurevych, 2019, 2020; Thakur et al., 2021) are examples of such approaches. As for contextual embeddings, these models are often built using DNNs for sequence processing.

Non-parametric sentence embeddings rely on pre-trained models for word embeddings or statistical methods to generate representations. Examples include averaging word embeddings, Smooth Inverse Frequency (SIF) WEIGHTING (Arora et al., 2017), and DYNAMAX (Zhelezniak et al., 2019), SFBOW (Muffo et al., 2021, 2023).

3 Model

DEF2VEC presents a novel approach to constructing word embeddings that exploits the structured information contained within dictionary definitions. The underlying principle of DEF2VEC involves the generation of term-document matrices from dictionary definitions, followed by LSA to yield semantically informative and extendable embeddings. In this section, we explain how to build the term-document matrix and how LSA is applied to this matrix to extract the embeddings. Additionally, we explain how the model can be extended with new embeddings without requiring any re-training, and, to conclude, we summarise the model capabilities.

3.1 Building the Term-Document Matrix

Given a vocabulary $\mathcal{V}$ of $|\mathcal{V}|$ terms (either words of multi-word expressions), each term in $\mathcal{V}$ is associated with one or more definitions extracted from linguistic resources. DEF2VEC constructs a term-document matrix $\mathbf{D}$, where each row corresponds to the *Term Frequency-Inverse Document Frequency*

(TF-IDF) representation of the definitions associated with a term. In the case of terms with multiple definitions (e.g., polysemous words), the TF-IDF vectors of individual definitions are averaged.

Mathematically, given a term $w \in \mathcal{V}$ represented as a *one-hot vector* $\mathbf{x} \in \mathbb{1}^{|\mathcal{V}|}$ (where $\mathbb{1} \equiv \{0,1\}$ and $\|\mathbf{x}\| = 1$) and its corresponding TF-IDF definition vector $\mathbf{y} \in \mathbb{R}^{|\mathcal{V}|}$, we establish the relationship defined in Equation (1).

$$\mathbf{y} = \mathbf{x} \cdot \mathbf{D} \tag{1}$$

Where $\mathbf{D} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is a *sparse matrix*, connecting terms to their definitions. $\mathbf{D}_i$, the i-th row of $\mathbf{D}$, is such that $\mathbf{D}_i = \mathbf{y}$ (supposing w is the i-th term in $\mathcal{V}$).

3.2 Latent Semantic Analysis

To distil semantic information and generate embeddings, Def2Vec applies LSA to the term-document matrix $\mathbf{D}$. LSA is de facto reduced (or truncated) *Singular Value Decomposition* (SVD), a method for matrix factorisation.

The term-document matrix $\mathbf{D}$ is factorised as reported in Equation (2). The process is represented in Figure 1.

$$\mathbf{D} \simeq \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^{\top} \tag{2}$$

Here, $\mathbf{\Sigma} \in \mathbb{R}^{d \times d}$ is a diagonal matrix with the singular values and $\mathbf{U} \in \mathbb{R}^{|\mathcal{V}| \times d}$ and $\mathbf{V} \in \mathbb{R}^{|\mathcal{V}| \times d}$ are matrices containing the left and right singular vectors, respectively. d is the desired embedding dimensionality, a tunable hyperparameter of the Def2Vec model.

3.3 Extensibility and Reconstruction

The significance of Def2Vec lies in its extensibility. SVD decomposition yields embeddings as rows of the matrix $\mathbf{U}$. However, the right singular vectors in $\mathbf{V}$ and the singular values in $\mathbf{\Sigma}$ can be exploited to generate embeddings from the TF-IDF representation of a term's definition as presented by Equation (3):

$$\mathbf{x} \cdot \mathbf{U} = \mathbf{U}_i = \mathbf{u} \simeq \mathbf{y} \cdot \mathbf{V} \cdot \mathbf{\Sigma}^{-1} \tag{3}$$

Both processes of embedding fetching from $\mathbf{U}$ and the embedding reconstruction from $\mathbf{V}$ and $\mathbf{\Sigma}$ are visualised in Figure 2.

This approach makes the embeddings extensible, enabling adding new terms without retraining the entire model. The downside of the approach is the small reconstruction error the truncation introduces after the SVD process. However, neural networks, which operate based on these embeddings, are expected to remain robust to the slight variations introduced by the reconstruction process (as we show in our evaluation).

Table 1: Comparison of vocabulary size and data set size of the considered word embedding models.

Model	Vocabulary size $\times 10^9$	No. training tokens $\times 10^9$
Def2Vec	0.76	0.05
Word2Vec	3	100
GloVe	2	840
FastText	2.19	600

3.4 Robustness and Quality of Representations

Def2Vec's embeddings benefit from the semantic richness of dictionary definitions while preserving the efficiency of static embeddings. This model leverages LSA's decomposition to capture latent semantic relationships within definitions, yielding embeddings that demonstrate semantic coherence even in the presence of noise and variation inherent in textual definitions.

In summary, Def2Vec introduces a novel methodology that combines the interpretability and extensibility of static embeddings with the rich semantic information present in dictionary definitions. We realised our implementation of Def2Vec using *Scikit-Learn* (Pedregosa et al., 2012), which offer utilities for TF-IDF vectorisation and SVD.

4 Data

The foundation of the Def2Vec model lies in the use of the Wiktionary[1] as a rich source of linguistic information. Wiktionary, a project by the *Wikipedia Foundation*, offers a comprehensive dictionary encompassing various languages, providing definitions, pronunciations, etymologies, and more for a wide array of terms. To construct the Def2Vec data set, we "mined" the English-language instance of the Wiktionary, extracting definitions to form the basis of our semantic representations.

To the end of this work, we used the dump file of the English Wiktionary from September 2020[2].

[1] Website: https://en.wiktionary.org/wiki/Wiktionary:Main_Page.

[2] The latest dumps of the Wikitionary are available

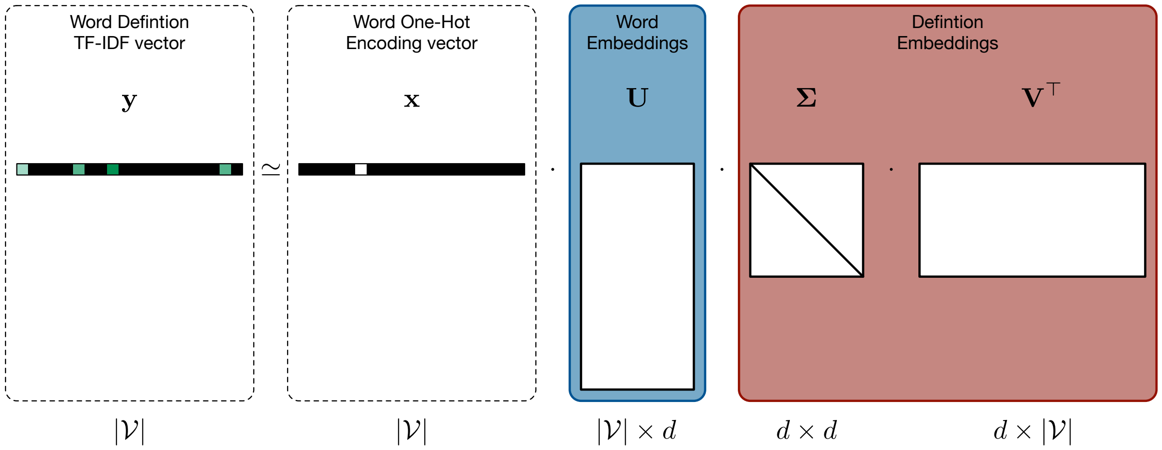


Figure 1: DEF2VEC term-document matrix decomposition.

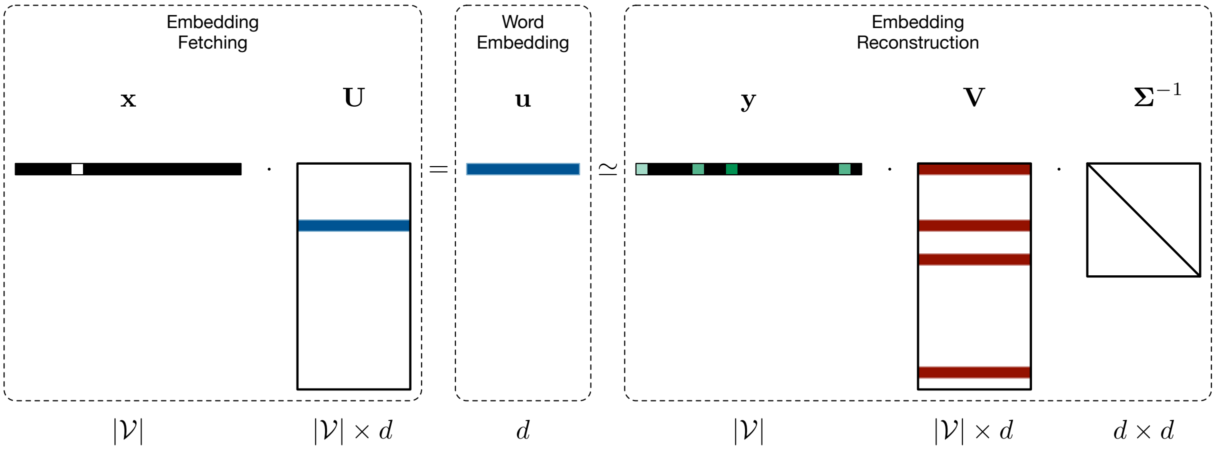


Figure 2: DEF2VEC embedding fetching and reconstruction processes.

The XML structure of the dump file consists of individual pages, each corresponding to a dictionary entry. A page includes a title, which is the term being defined, and a text section encapsulating the various elements of the entry, such as definitions, examples, synonyms, and more.

We cleaned and processed the data to generate a cohesive data set suitable for training the DEF2VEC model. We filtered out non-English entries and definitions associated with multiple languages, retaining only English ones. Additionally, we removed formatting tags, comments, and extraneous information, focusing solely on the textual content relevant to our work.

Each definition is preceded by the symbol #, which we removed during the parsing process. Our parser also excluded definitions marked with the label rfdef, indicating that the definitions did not exist on the corresponding WIKTIONARY web page. We further addressed links to other WIKTIONARY pages, Wikipedia pages, or appendices, ensuring that only relevant words were retained.

We made distinctions between *locutions* (multi-word expressions) and *proper nouns*. While generic locutions were excluded, locutions related to proper nouns were retained. Proper nouns carry distinct semantic significance and enrich the contextual understanding of terms. The data set consists of approximately 764,595 tokens and 1,023,372 definitions. Additionally, it comprises 12,903 locutions. Notably, the data set includes 39,251 tokens containing punctuation, allowing the model to capture the nuances of language even in the presence of punctuated terms. Compared to other word embedding models, ours is less “data-hungry” as highlighted in Table 1.

By leveraging the structured information in WIKTIONARY, we constructed a comprehensive data set that serves as the foundation for training the DEF2VEC model. The subsequent sections delve into the architecture of DEF2VEC and its

at the following link: https://dumps.wikimedia.org/enwiktionary/.

Table 2: Main statistics of the benchmark data sets.

Bechmark	Split	No. samples	Avg. no. tokens
CoNLL-2003	**Train**	14,041	14.5
	Val.	3250	15.8
	Test	3453	13.5
	Total	20,744	14.5
STS	**Train**	5749	22.8
	Val.	1500	26.4
	Test	1379	22.6
	Total	8628	23.4

Table 3: Fraction of sentences containing tokens removed for the reconstruction evaluation.

Bechmark	Split	Faction of sentences [%]
CoNLL-2003	**Train**	40.7
	Val.	40.5
	Test	42.6
STS	**Train**	46.5
	Val.	50.0
	Test	60.2

performance across various tasks, illustrating its unique approach to word embeddings.

5 Evaluation

This comprehensive section presents our evaluation strategy for the DEF2VEC model. We describe the selected benchmarks, the evaluation approach, and the baselines we used for comparison.

5.1 Benchmarks

Our evaluation benchmarks encompass diverse linguistic tasks, providing a comprehensive understanding of DEF2VEC's performance.

We employed the CoNLL-2003 data set (Tjong Kim Sang and De Meulder, 2003) for sequence labelling tasks: namely *Part-of-Speech* (POS) tagging, *Named Entity Recognition* (NER) and *chunking* (CHUNK). The CoNLL-2003 data set was proposed as NER benchmark in the *CoNLL* conference (Daelemans and Osborne, 2003) and provides tags in *BIO* format for POS, NER, and CHUNK tasks. The data set, divided into training, validation, and test splits, facilitated an evaluation of DEF2VEC's capabilities in capturing linguistic structures and semantic nuances. Each sample is a pre-tokenised sentence (the input); each token of the sentence has its reference labels (the target output).

For sentence similarity, we turned to the Semantic Textual Similarity (STS) data set (Cer et al., 2017), evaluating DEF2VEC's ability to capture semantic relationships. The STS Benchmark comprises a selection of the English corpora used in organised in the context of the *SemEval* challenges between 2012 and 2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016). The selection of corpora composing the data set includes text from image captions, news headlines, and user forums. Each sample in the corpus comprises a pair of sentences (the input) and their similarity score (the target output).

5.2 Approach

Our evaluation approach involves a two-pronged strategy: assessing embedding quality and reconstruction effectiveness. We trained Convolutional Neural Networks (CNNs) tailored to the tasks to evaluate embedding quality. These CNNs featured essential layers to ensure robust evaluations while mitigating overfitting risks:

- Dropout Layers: We introduced dropout layers with a 10% dropout probability, serving as regularisation mechanisms during training.
- Convolutional Layers: Our architecture included a convolutional layer with a kernel width of 5 embeddings, using the *Gaussian Error Linear Unit* (GELU) activation function for non-linearity.
- Additional Dropout: An extra dropout layer for further regularisation, with a 10% probability, followed the convolutional layers.

For sequence labelling tasks (POS, NER, CHUNK), our approach incorporated a linear layer to map input vectors to *logit* scores. Applying sequence-wise softmax operations yielded label probabilities. This enabled an in-depth evaluation of DEF2VEC's capacity to capture syntactic and semantic features across various linguistic tasks.

We utilised the *Semantic Textual Similarity* (STS) data set to train parametric sentence embedding models. We generated the sentence embeddings by employing a siamese network architecture with attention pooling. We computed the *cosine similarity* of these embeddings to quantify the sentence similarity. This benchmark evaluated Def2Vec's appropriateness to capture nuanced semantic relationships at sentence level.

The choices in the neural network architectures were mainly guided by the reference work of Collobert et al. (2011), where word embeddings were first used to improve results on different NLP tasks.

In addition to assessing embedding quality, we examined Def2Vec's reconstruction capabilities. We pruned the Wiktionary data set, removing words with frequencies of 10 or fewer occurrences. Subsequently, we retrained Def2Vec on the remaining 30,174 terms and their definitions.

We employed this reduced model to reconstruct embeddings for words in the benchmark samples lacking embeddings. We reported the statistics on the affected samples in Table 3. These reconstructed embeddings were generated from their Wiktionary definitions.

5.3 Baselines

Throughout all benchmark tasks, we conducted extensive comparisons between Def2Vec (D2V) and established word embedding models: Word2Vec (W2V), GloVe (GV), and FastText (FT). These comparisons allowed us to gauge Def2Vec's performance against widely recognised methods.

This comprehensive evaluation approach, supplemented by a thorough assessment of reconstruction capabilities, is the foundation for our analysis of Def2Vec's performance in the subsequent results section. By investigating both embedding quality and reconstruction effectiveness, we aim to understand Def2Vec's capabilities in capturing and representing semantic information within dictionary definitions.

6 Results

This section presents and discusses the results of our proposed Def2Vec model across various linguistic tasks.

6.1 Sequence Labelling Tasks

Tables 4 to 6 showcase the classification results of Def2Vec, along with comparisons to established embedding models, on the CoNLL-2003 data set for the POS, NER, and CHUNK tasks, respectively. Across all tasks, Def2Vec demonstrates competitive performances.

In the POS task, Def2Vec achieves accuracy scores of 73.64% (Validation) and 72.42% (Test), showing its proficiency in capturing syntactic information. It clearly outperforms Word2Vec, but is still distant from GloVe and FastText.

Table 4: Classification results on the POS task from CoNLL-2003.

Model	**Split**	**Metric [%]**				
		Acc.	**Prec.**	**Rec.**	**F$_1$**	**AUC**
D2V	**Val.**	73.64	85.68	73.64	77.62	95.84
	Test	72.42	85.41	72.42	76.55	94.63
W2V	**Val.**	64.13	85.70	64.13	70.20	94.99
	Test	60.01	85.92	60.01	67.45	94.07
GV	**Val.**	82.53	90.49	82.53	85.43	97.94
	Test	82.38	90.79	82.38	85.51	97.86
FT	**Val.**	80.27	90.47	80.27	83.72	97.81
	Test	79.90	90.31	79.90	83.30	97.73

Table 5: Classification results on the NER task from CoNLL-2003.

Model	**Split**	**Metric [%]**				
		Acc.	**Prec.**	**Rec.**	**F$_1$**	**AUC**
D2V	**Val.**	73.89	99.31	73.89	83.89	97.24
	Test	71.98	99.28	71.98	83.09	96.28
W2V	**Val.**	75.00	99.21	75.00	84.50	96.52
	Test	73.31	99.25	73.31	83.95	95.44
GV	**Val.**	91.80	99.58	91.80	95.34	99.29
	Test	90.52	99.47	90.52	94.60	99.21
FT	**Val.**	90.30	99.57	90.30	94.51	99.20
	Test	89.32	99.47	89.32	93.93	99.12

Table 6: Classification results on the CHUNK task from CoNLL-2003.

Model	**Split**	**Metric [%]**				
		Acc.	**Prec.**	**Rec.**	**F$_1$**	**AUC**
D2V	**Val.**	77.79	86.81	77.79	81.34	94.37
	Test	77.69	86.56	77.69	81.45	93.07
W2V	**Val.**	66.12	82.97	66.12	71.35	90.28
	Test	64.94	82.19	64.94	71.00	87.91
GV	**Val.**	80.09	89.86	80.09	84.09	95.18
	Test	79.43	89.20	79.43	83.51	94.49
FT	**Val.**	82.38	90.21	82.38	85.60	95.03
	Test	82.28	89.63	82.28	85.39	94.55

For NER, Def2Vec achieves 73.89% (Validation) and 71.98% (Test) accuracy. While GloVe and FastText yields the highest accuracy, Def2Vec remains competitive in precision, recall, F$_1$, and AUC. Results against Word2Vec are still comparable on all metrics.

Table 7: Spearman correlation score on the different subsets of the STS benchmark.

Model	Split	Spearman correlation [%]			
		Subset			Total
		Caption	Forum	News	
D2V	**Val.**	76.27	30.17	60.84	69.98
	Test	75.52	42.68	57.43	63.72
W2V	**Val.**	83.33	49.61	63.22	77.67
	Test	81.57	52.46	59.18	69.45
GV	**Val.**	83.45	55.96	66.60	78.37
	Test	80.17	53.49	63.16	69.00
FT	**Val.**	86.08	58.95	70.08	81.14
	Test	83.27	59.92	61.48	72.49

In the CHUNK task, Def2Vec achieves an accuracy of 77.79% (Validation) and 77.69% (Test), consistently competing with established methods and again outperforming Word2Vec.

Across these tasks, Def2Vec demonstrates its proficiency in capturing syntactic and semantic features, effectively supporting sequence labelling tasks. Def2Vec consistenstly performs better or similar to Word2Vec embeddings. However, Def2Vec only gets close, but never outperforms more sophisticated embeddings like GloVe and FastText.

6.2 Sentence Similarity Benchmark

The Spearman correlation scores for the STS benchmark are shown in Table Table 7. Here, we can see that Def2Vec's performance in capturing semantic relationships among sentence pairs is satisfactory.

For the validation subset, Def2Vec achieves a Spearman correlation score of 69.98% (Total), with a string drop on the Forum subset (30.17%). All the other models share this drop, but it is not equally remarkable.

In the test subset, Def2Vec obtains a correlation of 63.72% (Total). Differently from sequence labelling tasks, the gap with the validation results is more significant, but, again, it is a behaviour similar to that of all the baselines. Differently from the other models, the gap (absolute or relative) between validation and test is lower, hinting at higher robustness of the sentence embeddings.

While outperformed by the other model in all subsets, Def2Vec maintains competitive performance and robustly captures semantic information, yielding overall Spearman correlations $> 60\%$.

Table 8: Classification results on the CoNLL-2003 tasks of the reconstructed Def2Vec embeddings.

Task	Split	Metric [%]				
		Acc.	Prec.	Rec.	F_1	AUC
POS	**Val.**	74.35	86.53	74.35	78.40	96.11
	Test	73.38	86.35	73.38	77.54	94.99
NER	**Val.**	74.19	99.32	74.19	84.09	97.23
	Test	72.28	99.29	72.28	83.30	96.37
CHUNK	**Val.**	77.84	86.97	77.84	81.47	94.44
	Test	77.84	86.61	77.84	81.56	93.08

Table 9: Differences between the classification scores on the CoNLL-2003 tasks of the reconstructed Def2Vec word embeddings and the original ones.

Task	Split	ΔMetric [%]				
		Acc.	Prec.	Rec.	F_1	AUC
POS	**Val.**	0.71	0.85	0.71	0.78	0.27
	Test	0.95	0.94	0.95	0.98	0.36
NER	**Val.**	0.29	0.01	0.29	0.20	−0.01
	Test	0.30	0.01	0.30	0.21	0.08
CHUNK	**Val.**	0.05	0.16	0.05	0.13	0.07
	Test	0.15	0.05	0.15	0.12	0.01

Table 10: Spearman correlation score on the STS benchmark of the reconstructed Def2Vec embeddings.

Task	Split	Spearman correlation [%]			
		Subset			Total
		Caption	Forum	News	
STS	**Val.**	75.93	34.15	61.74	70.73
	Test	73.19	43.05	57.47	62.57

Table 11: Differences between the Spearman correlation scores on the STS benchmark of the Def2Vec reconstructed word embeddings and the original ones.

Task	Split	ΔSpearman correlation [%]			
		Subset			Total
		Caption	Forum	News	
STS	**Val.**	−0.34	3.98	0.90	0.75
	Test	−2.32	0.36	0.05	−1.15

6.3 Reconstruction Capabilities

We evaluate Def2Vec's reconstruction capabilities using the CoNLL-2003 data set and the STS benchmark with reconstructed embeddings. Tables 8 and 10 depict the results of the models trained with the reconstructed embeddings, and

Tables 9 and 11 highlight the differences between the results obtained by the original DEF2VEC (trained on all the WIKITIONARY data) and the reconstructed embeddings.

For sequence labelling tasks (POS, NER, CHUNK), DEF2VEC's reconstructed embeddings exhibit slightly lower accuracy, precision, recall, and F_1 than original embeddings. However, the differences are generally marginal, showcasing the effectiveness of the reconstruction process.

Reconstructed embeddings exhibit varying performance across subsets in the STS benchmark. Some subsets show minor decreases in Spearman correlation scores, while others display improvements. Notably, the Forum subset's performance sees improvement in correlation scores, indicating the effectiveness of the reconstruction process in capturing specific nuances.

6.4 Model Discussion

DEF2VEC consistently showcases competitive performance across sequence labelling tasks and sentence similarity benchmarks. While its reconstructed embeddings exhibit slight variations in performance, the overall impact remains limited. This highlights DEF2VEC's robustness and potential to effectively capture and represent semantic information.

In conclusion, the DEF2VEC model presents a promising approach for learning word embeddings from dictionary definitions. Its semantic embedding quality and reconstruction capabilities demonstrate its utility in various linguistic tasks, making it a suitable alternative for advancing natural language understanding tasks in diverse applications.

7 Conclusion

In this study, we introduced DEF2VEC, a novel approach for learning word embeddings by leveraging dictionary definitions. DEF2VEC capitalises on the rich semantic information present in definitions to create embeddings that capture syntactic and semantic features. Through a comprehensive evaluation, we demonstrated the efficacy of DEF2VEC across various linguistic tasks, showcasing its ability to compete with established embedding models.

In the sequence labelling tasks of POS, NER, and CHUNK, DEF2VEC exhibited competitive accuracy, precision, recall, and F_1, illustrating its effectiveness in capturing linguistic nuances. Additionally, the model's performance on the STS benchmark reflected its capability to discern semantic relationships among sentence pairs, highlighting its utility in gauging semantic similarity across different contexts.

Moreover, we explored the dynamic extensibility of DEF2VEC, evaluating its ability to reconstruct embeddings of out-of-vocabulary words from their definitions. The results indicated that while the reconstructed embeddings displayed slight variations in performance, the overall impact remained limited, underscoring the robustness of the approach.

Our work opens for several future developments:

- Extending DEF2VEC to incorporate sub-word information, such as morphemes or character-level embeddings, could enhance its ability to capture finer linguistic nuances and improve its performance on tasks involving rare or out-of-vocabulary words.
- Adapting DEF2VEC to other languages can uncover cross-lingual variations in lexical semantics and offer insights into the universality of the approach. This could lead to the creation of embeddings that facilitate multilingual natural language processing tasks.
- Exploring different corpora than the Wikitionary could help assess the effect of the training data and identify better data sources.
- Conducting a more extensive evaluation of DEF2VEC on a broader array of linguistic tasks, such as syntactic parsing and semantic role labelling, could further validate its robustness and versatility.
- Utilising DEF2VEC embeddings as initialisation for training deep contextual models like BERT, GPT, or their successors could enhance language understanding and generation capabilities, potentially contributing to advancements in various natural language processing applications.

In conclusion, DEF2VEC introduces a novel perspective on word embedding learning that exploits dictionary definitions to produce embeddings with both syntactic and semantic information, which are also extensible. Its competitive performance across tasks and the potential for future extensions make it a promising candidate for enhancing the landscape of word embeddings and advancing natural language understanding.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91, Dublin, Ireland. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA. Association for Computational Linguistics.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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.

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*.

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.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Walter Daelemans and Miles Osborne, editors. 2003. *Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003*. ACL.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Scott C Deerwester, Susan T Dumais, George W Furnas, Richard A Harshman, Thomas K Landauer, Karen E Lochbaum, and Lynn A Streeter. 1989. Computer information retrieval using latent semantic structure. US Patent 4,839,853.

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.

Jeffrey L. Elman. 1990. Finding structure in time. *Cogn. Sci.*, 14(2):179–211.

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

Marjan Hosseini, Alireza Javadian Sabet, Suining He, and Derek Aguiar. 2023. Interpretable fake news detection with topic and deep variational models. *Online Soc. Networks Media*, 36:100249.

Dan Jurafsky and James H. Martin. 2023. Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition (3rd edition). Draft.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3294–3302.

Qi Liu, Matt J. Kusner, and Phil Blunsom. 2020a. A survey on contextual embeddings. *CoRR*, abs/2003.07278.

Zhiyuan Liu, Yankai Lin, and Maosong Sun. 2020b. *Representation Learning for Natural Language Processing*. Springer.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.

Matteo Muffo, Roberto Tedesco, Licia Sbattella, and Vincenzo Scotti. 2021. Static fuzzy bag-of-words: a lightweight and fast sentence embedding algorithm. In *Proceedings of the Fourth International Conference on Natural Language and Speech Processing (ICNLSP 2021)*, pages 73–82, Trento, Italy. Association for Computational Linguistics.

Matteo Muffo, Roberto Tedesco, Licia Sbattella, and Vincenzo Scotti. 2023. *Static Fuzzy Bag-of-Words: Exploring Static Universe Matrices for Sentence Embeddings*, pages 191–211. Springer International Publishing, Cham.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. 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 528–540, New Orleans, Louisiana. Association for Computational Linguistics.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2012. Scikit-learn: Machine learning in python. *CoRR*, abs/1201.0490.

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

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. *OpenAI blog*, 1(11):12.

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

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

Radim Řehůřek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of LREC 2010 workshop New Challenges for NLP Frameworks*, pages 45–50, University of Malta, Valletta, Malta. University of Malta.

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

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon

Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 296–310, Online. Association for Computational Linguistics.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Thomas Wang, Adam Roberts, Daniel Hesslow, Teven Le Scao, Hyung Won Chung, Iz Beltagy, Julien Launay, and Colin Raffel. 2022. What language model architecture and pretraining objective works best for zero-shot generalization? In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 22964–22984. PMLR.

Vitalii Zhelezniak, Aleksandar Savkov, April Shen, Francesco Moramarco, Jack Flann, and Nils Y. Hammerla. 2019. Don't settle for average, go for the max: Fuzzy sets and max-pooled word vectors. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

A Pre-trained embeddings

The pre-trained embeddings for English used in the experiments are accessible in *Gensim*-compatible format (Řehůřek and Sojka, 2010). The embeddings can be downloaded from the *GitHub* repository[3].

[3] https : / / github.com / IreneMorazzoni / def_2_vec_irene

Exploring Hybrid Linguistic Features for Turkish Text Readability

Ahmet Yavuz Uluslu
Universität Zürich
ahmetyavuz.uluslu@uzh.ch

Gerold Schneider
Universität Zürich
gschneid@cl.uzh.ch

Abstract

This paper presents the first comprehensive study on automatic readability assessment of Turkish texts. We combine state-of-the-art neural network models with linguistic features at lexical, morphosyntactic, syntactic and discourse levels to develop an advanced readability tool. We evaluate the effectiveness of traditional readability formulas compared to modern automated methods and identify key linguistic features that determine the readability of Turkish texts.

1 Introduction

Automatic Readability Assessment (ARA) is an important task in computational linguistics that aims to automatically determine the level of difficulty of understanding a written text, which has implications for various fields, such as healthcare, education, and accessibility (Vajjala, 2022). In the healthcare sector, medical practitioners can use ARA tools to ensure patient information and consent forms are easily understandable (Ley and Florio, 1996). In the field of education, teachers and learners alike can benefit from ARA systems to adapt materials to the appropriate language proficiency level (Kintsch and Vipond, 2014). The appropriate readability of technical reports and other business documents is critical to ensure that the intended audience can fully understand the content and can make informed decisions (Bushee et al., 2018). In areas such as cyber-security, readability is particularly important as it can impact response time to risk closures and case materials (Smit et al., 2021).

The task of assessing readability presents challenges, particularly when dealing with large corpora of text. Although manual linguistic analysis by domain experts provides valuable insights, it is time-consuming, costly and subject to individual interpretation, which can lead to variability and subjectivity in the annotation results (Deutsch et al., 2020). Recent research in the field has focused on developing automated methods for extracting linguistic predictors and training models for readability assessment. Despite these crucial applications and developments, the readability efforts in Turkish have largely been confined to traditional readability formulas, such as Flesch-Kincaid (Kincaid et al., 1975) and its adaptations (Ateşman, 1997; Bezirci and Yilmaz, 2010; Çetinkaya, 2010). Several previous studies have pointed out the shortcomings of these formulas (Feng et al., 2010, 2009). They typically rely on superficial text features such as sentence length and word length. The integration of complex morphological, syntactic, semantic, and discourse features in modern ARA approaches offers the possibility of significantly improving the current readability studies in Turkish. In this paper, we present the first ARA study for Turkish. Our study combines traditional raw text features with lexical, morpho-syntactic, and syntactic information to create an advanced readability assessment tool for Turkish. We demonstrate the effectiveness of our tool on a new corpus of Turkish popular science magazine articles, published for different age groups and educational levels. Our study aims to contribute to the development of automated tools for accessibility, educational research, and language learning in Turkish.

The rest of the paper is organized as follows. In Section 2, we review related work on readability assessment and machine learning-based approaches. In Section 3, we describe our corpus and the linguistic features used in our study. In Section 4, we present the results of our experiments and analyze the effectiveness of our tool. Finally, in Section 5, we conclude our research and discuss future directions.

2 Previous Work

The research of quantifying text readability, or the ease with which a text can be read, has a history spanning over a century (DuBay, 2007). Initial research was centered on the creation of lists of difficult words and readability formulas such as Flesch Reading Ease (Flesch, 1948), Dale-Chall readability formula (Dale and Chall, 1948), Gunning FOG Index (Gunning, 1969) and SMOG (Mc Laughlin, 1969). These formulas are essentially simple weighted linear functions that utilize easily measurable variables such as word and sentence length, as well as the proportion of complex words within a text. Initially developed for the English language, the Flesch Reading Ease formula required recalibration for its application to Turkish, a task undertaken by Ateşman (1997). However, a significant obstacle in its adoption was Atesman's failure to disclose the statistical variables used in the recalibration process. This gap was later addressed in the work of Çetinkaya (2010), which also assigned appropriate grade levels, thus facilitating its practical use in the Turkish educational context. Not long after the adaptation, Bezirci and Yilmaz (2010) introduced an important refinement, akin to the approach taken in the SMOG formula. They propose that features based on polysyllabic words and the total number of syllables present in the document provide distinct indications of text complexity. Accordingly, they included the counts of polysyllabic words (those with 3-, 4-, and 5+ syllables). Sönmez (2003) encountered inconsistencies when applying the Gunning FOG Index to Turkish texts which led to the development of their adaptation. The limitations are mainly due to the subjective nature of the formula in identifying complex words and concepts, which contrasts with other formulas that use easier-to-identify criteria such as syllable counts.

Readability assessment has found practical applications in several areas in Turkish, particularly in the fields of medicine and education. For instance, researchers have used the Flesch-Kincaid and Atesman readability formulae to assess the readability of anaesthesia consent forms in Turkish hospitals, which led to valuable insights into how these documents could be optimised for better comprehension (Boztas et al., 2017; Boztaş et al., 2014). In the realm of education, readability studies have been employed to evaluate the complexity of textbooks, thereby ensuring that these crucial learning materials are appropriate for the targeted student age group. For example, research has been conducted to determine the readability levels of Turkish tales in middle-school textbooks, providing insights that could potentially enhance the quality of education by aligning learning materials with students' comprehension abilities (Turkben, 2019; Tekşan et al., 2020; Guven, 2014). While traditional readability formulas have significantly contributed to the field of readability assessment, they are not without their limitations. They often rely heavily on surface-level text features, such as word and sentence length, and fail to account for deeper linguistic and cognitive factors that influence readability (Collins-Thompson, 2014).

Readability formulae have inherent limitations that can affect their accuracy and applicability. Given the unique phonetic attributes, sentence formation patterns, and mean syllable length in each language, each language requires its own calibrated readability formula. The validity of studies employing readability formulae calibrated for the English language to evaluate texts in other languages remains questionable. In practice, applying an English-calibrated formula to Turkish texts may result in an overestimation of readability levels. Indeed, most studies that have used this approach have reported inflated levels of readability requirements (Akgül, 2019; Akgül, 2022) without accounting for the issues of calibration. Furthermore, the evolution of language over time may necessitate periodic re-calibration of these formulas (Lee and Lee, 2023). As language trends evolve and new words and phrases become more common, readability formulas must adapt to remain accurate and relevant. Previous research shows that traditional readability metrics perform unreliably when applied to non-traditional document types such as web pages (Petersen and Ostendorf, 2009).

Traditional readability formulas, despite their extensive use, have been criticised for their lack of wide linguistic coverage (Feng et al., 2009, 2010). These formulas predominantly focus on superficial text features, largely ignoring other linguistic aspects that significantly contribute to text readability. Factors such as syntactic and semantic complexity, discourse structure, and other linguistic branches recognised by (Collins-Thompson, 2014) which are integral to comprehending a text, remain largely unaccounted for in these traditional models. This narrow linguistic focus can lead to inaccuracies in readability assessment, especially when applied to languages or texts with diverse linguistic structures. These scores are relative measures of readability that should be interpreted in the context of the text's overall features and the target audience's reading ability. They are not absolute measures and treating them as such can result in a misunderstanding of the text's actual readability.

Practitioner errors in applying readability formulas often stem from methodological shortcomings and misinterpretations (Wang et al., 2013). The requirement of considerable text sample sizes for traditional measures introduces another impediment, even though the theoretical minimum size for a text sample has yet to be conclusively established. A common methodological error is the inappropriate sampling of text. Some studies might only consider a limited section of a text, such as the first 100 words, leading to skewed results, especially in scientific texts where complexity often increases later in the document. Similarly, the selective assessment of text sections that do not accurately mirror the overall complexity of the text, like focusing solely on the introduction or conclusion, can misrepresent the readability level.

In recent years, research in ARA has shifted from traditional linear models, which use simple metrics such as word and sentence length to estimate the reading level of a text, to fine-grained features (Collins-Thompson, 2014). These features often include output of machine learning models trained on a combination of word counts, lexical patterns, discourse analysis, morphology, and syntactic structures. There has been an emerging trend toward using neural models for ARA. These models have demonstrated the capacity to implicitly capture the previously mentioned features without the need for manually-defined feature extraction (Jawahar et al., 2019). Martinc et al. (2021) and Imperial (2021) experimented with contextual embeddings of BERT (Devlin et al., 2019) for the readability assessment task, achieving par or better results than feature-based approaches. However, both studies omitted cross-domain evaluation, leading to uncertainty about the extent to which language models rely on topic and genre information, as opposed to readability. Other studies have further explored various strategies to integrate linguistic features with transformer models, promoting a fusion of traditional and neural approaches (Lee et al., 2021; Deutsch et al., 2020). The state-of-the-art results are currently being achieved by hybrid models that ensemble linguistic features with transformer-based models, highlighting the combined strength of traditional and modern approaches.

3 Corpus

Most widely used readability corpora include One Stop English (OSE) (Vajjala and Lučić, 2018), the WeeBit corpus (Vajjala and Meurers, 2012) and the Newsela corpus (Xu et al., 2015). While the majority of these benchmark datasets and corpora are predominantly available in English, there is a growing interest in the development of readability corpora in other languages. In the context of low-resource languages, limited access to digital text resources necessitates reliance on conventional learning materials, such as classroom materials and textbooks. There are currently no existing readability corpora available for Turkish.

3.1 TUBITAK PopSci Magazine Readability Corpus

Our corpus was constructed using popular science articles from TUBITAK Popular Science Magazines [1] spanning the period 2007 to 2022. The articles are openly published and made available for non-commercial redistribution and research purposes. We selected 2250 articles from three magazines, each catering to readers of different age groups. These magazines include Meraklı Minik (for ages 0-6), Bilim Çocuk (for ages 7+), and Bilim ve Teknik (for ages 15+). Accordingly, we consider the articles from these magazines as elementary, intermediate, and advanced level reading material. Our corpus is non-parallel and encompasses a diverse range of topics, including instructions for laboratory experiments and brief articles about recent scientific discoveries. This characteristic is similar to that of the WeeBit corpus (Vajjala and Meurers, 2012), which also includes articles from various topics and resources. Given that the articles in our corpus are written by experts and specifically tailored for distinct age groups, it can be appropriately regarded as an 'expert-annotated' corpus. We used a off-the-shelf pdf-to-text converter to extract the relevant article text and manually corrected the articles to ensure the conversion accuracy of Turkish characters and the layout integrity. Table 1 displays descriptive statistics for the finalized corpus.

As expected, the advanced texts display a greater average length compared to the elementary texts. However, the high standard deviation values for each level indicate that other factors beyond text

[1] https://yayinlar.tubitak.gov.tr/

Level	Avg. Words	Std. Dev	Nr. of Articles
ELE	120.95	67.35	750
INT	154.99	93.57	750
ADV	327.08	187.54	750

Table 1: Descriptive Corpus Statistics

length may have a significant impact on determining the reading level of a given text.

We also performed a preliminary analysis on the three reading levels of the corpus using traditional formulae and showed the results in Table 2, presenting readability metrics Atesman, Cetinkaya-Uzun and Type-Token Ratio (TTR). Atesman and Cetinkaya readability scores decrease from one level to the next indicating that texts become more complex at higher reading levels. In contrast, the TTR score increases suggesting that texts become more diverse and less repetitive at higher reading levels. It should also be noted that the readability levels of the elementary-level articles in both formulas were not suitable for the intended age group and that the magazine's disclaimer states that certain articles may require the assistance of an adult or parent. Table 3 presents examples of articles representing each of the three reading levels.

Feature	ELE	INT	ADV
Atesman	66.06	59.73	42.32
Cetinkaya	39.31	36.62	29.81
TTR	0.65	0.71	0.76

Table 2: Readability features across reading levels

4 Linguistic Features

In this study, we explore five subgroups of linguistic features from our Turkish readability corpus: traditional or surface-based features, syntactic features, lexico-semantic features, morphological features, and discourse features. We employ spaCy v3.4.0 (Honnibal et al., 2020) with the pre-trained tr_core_news_trf model[2] for the majority of general tasks, including entity recognition, POS tagging, and dependency parsing. We use the Stanford Stanza parser version 1.5.0 (Qi et al., 2020) for constituency parsing.

4.1 Traditional Features (TRAD)

Traditional or surface-based features are commonly used to predict the readability of Turkish texts, and we also adopt them as a baseline for our study. Specifically, we extract 7 traditional features, including Turkish adaptations of well-known readability formulas such as Atesman and Cetinkaya-Uzun, as well as average numbers of words and syllables per document. As noted by (Bezirci and Yilmaz, 2010) in their evaluation of the Turkish readability formulae, the impact of the number of polysyllabic words on text complexity is different from that of the total number of syllables present in the text. Therefore, we also included the counts of polysyllabic words (3-, 4-, and 5+ syllables) as separate features in our analysis.

4.2 Syntactic Features (SYNX)

Syntactic properties have a significant impact on the overall complexity of a given text, which serves as an important indicator of readability. We extract an array of syntactic features that capture various dimensions of sentence structure.

Phrasal and dependency type features: Reading abilities are related to the ratios involving clauses in a text (Lu, 2010). We extract features based on noun and verb phrases at sentence and article levels. We integrate features based on the unconditional probabilities of their dependency-based equivalents (Dell'Orletta et al., 2011). These encompass various types of syntactic dependencies, including subject, direct object, and modifier, among others.

Parse tree depth features: The depth and structure of dependency trees in a text can reflect the level of sentence complexity. Following this principle, we extract the average and maximum depths of the constituency and dependency tree structures present in the text (Dell'Orletta et al., 2011).

Part-of-Speech features: Part-of-speech (POS) tags provide essential information about the syntactic function of words in sentences. Adapting the work of Tonelli et al. (2012) and Lee et al. (2021), we include features based on universal POS tag counts. Such features offer insights into the distribution and usage of different word categories, adding another layer of syntactic information.

4.3 Lexico-Semantic Features (LXSM)

Lexico-semantic features are a set of linguistic attributes that can reveal the complexity of a text's vocabulary. These features can be used to identify specific words or phrases that may pose difficulty or unfamiliarity to readers (Collins-Thompson, 2014).

[2] https://huggingface.co/turkish-nlp-suite/tr_core_news_trf

Reading Level	Example
Elementary	Burası bir doğa koruma merkezi. Burada annesi ve babası olmayan turna yavruları var. Merkezde çalışanlardan biri özel bir giysi giyip koluna bir turna kuklası geçirmiş. *(This is a nature conservation centre. There are crane chicks without a mother and father. One of the workers at the centre wears a special suit and a crane puppet on his arm.)*
Intermediate	Robotlar, insanların yaptığı işleri, onların yerine yapan karmaşık makinelerdir. Bu işleri yapmak için programlanırlar. Otomatik olarak ya da uzaktan kumanda edilerek belirli komutları yerine getirirler. *(Robots are complex machines that do the jobs that humans do, instead of them. They are programmed to do these jobs. They fulfil certain commands automatically or by remote control.)*
Advanced	Pek çok canlıda manyetik algının varlığı bilimsel olarak biliniyor. Bakteri, salyangoz, kurbağa ve ıstakoz gibi canlılar Dünya'nın manyetik alanını algılıyor, göçmen kuşlar ve deniz kaplumbağaları yönlerini bu sayede buluyor, köpekler eğitildiklerinde saklanmış çubuk mıknatısın yerini gösterebiliyor. *(The existence of magnetic perception in many living things is scientifically known. Bacteria, snails, frogs and lobsters can sense the Earth's magnetic field, migratory birds and sea turtles can navigate, and dogs can point out a hidden bar magnet when trained to do so.)*

Table 3: Example sentences for three reading levels

Lexical Variation features: Secondary language acquisition research has found a correlation between the diversity of words within the same Part-Of-Speech (POS) category and the lexical richness of a text (Vajjala and Meurers, 2012). We extract noun, verb, adjective, and adverb variations, which represent the proportion of the respective category's words to the total.

Type Token Ratio (TTR) features: TTR is a commonly used metric to quantify lexical richness and has been widely employed in readability assessment studies. We compute five distinct variations of TTR from (Vajjala and Meurers, 2012). The standard TTR variations of a text sample are susceptible to the text length, which can introduce bias in the readability assessment. To address this limitation, we also consider the Moving-Average Type–Token Ratio (MATTR) (Covington and McFall, 2010). The MATTR mitigates the length-dependency issue by calculating the TTR score within a moving window across the text.

Psycholinguistic features: We adopted word frequencies obtained from the Turkish psycholinguistic database created by Acar et al. (2016). This resource was built from transcriptions of children's speech and corpora of children's literature, thus containing words commonly acquired during early development. It also includes words typically acquired during adulthood from a standard corpus. We extracted the average word and sentence frequency for both early and late-acquired words. We calculate features based on the average log10 values similar to the SubtlexUS corpus (Brysbaert and New, 2009).

Word Familiarity features: Familiarity with specific words can greatly affect readability. Based on prior work on Italian (Dell'Orletta et al., 2011) and French (François and Fairon, 2012) readability studies, we assessed the vocabulary composition of the articles using a reference list of 1700 basic words essential for achieving elementary reading proficiency in Turkish. This list, a combination of the first 1200 words taught to children aged 0-6 (Keklik, 2010) and a set of essential words from an open-access textbook[3] for learning Turkish, provides a benchmark for vocabulary familiarity. We calculated the percentage of unique words (types) in the text based on this reference list, performed on a lemma basis.

4.4 Morphological features (MORPH)

Morphological complexity plays a significant role in readability assessment, particularly in languages that are morphologically richer than English such as German (Hancke et al., 2012) and Basque (Gonzalez-Dios et al., 2014). In our study, we integrate the Morphological Complexity Index (MCI) from Brezina and Pallotti (2019). The MCI cap-

[3] https://www.turkishtextbook.com/most-common-words/

tures the variability of morphological exponents of specific parts-of-speech within a text by comparing word forms with their stems. We calculate MCI features for verbs, nouns, and adjectives, considering different sample sizes and sampling techniques with and without repetition. MCI has been leveraged in cross-lingual readability assessment frameworks, proving its applicability across languages with varying morphological structures (Weiss et al., 2021). However, these studies have not explored agglutinative languages such as Turkish and Hungarian.

4.5 Discourse features (DISCO)

The final group of features we examine are entity density features. The presence and frequency of entities within a text can significantly impact the cognitive load required for comprehension. Entities often introduce new conceptual information, thereby increasing the burden on the reader's working memory. This relationship between entities and readability was previously shown by Feng et al. (2009, 2010).

5 Experiments

We experiment with four different setups: trad-baseline (non-neural model with shallow features), modern-baseline (non-neural model with linguistic features), neural (pretrained transformer models), and hybrid (modern-baseline + neural). We use 10-fold cross-validation (10FCV) and evaluate our models using standard metrics such as accuracy, precision, recall, and macro F1-score. Specifically, we choose traditional learning algorithms such as Logistic Regression, Support Vector Machines, Random Forest and XGBoost as our baseline models. We perform a randomised search to explore a reasonable range of hyper-parameter values. We apply a grid search to identify the optimal combination of hyper-parameter values within this range.

5.1 Non-Neural Models with Linguistic Features

Given the lack of available baselines for the readability task in Turkish, our first objective is to establish a baseline for the readability task. This baseline (trad-baseline) is designed to be on par with traditional readability formulas and is reliant on shallow linguistic features such as sentence and word lengths. By establishing this baseline, we are effectively creating a benchmark that allows for meaningful comparison between the traditional readability formulas, which are the only available methods in readability assessment for Turkish. We expand our feature set and include a more diverse set of linguistic feature groups (modern-baseline). We are interested in the performance of individual features, but we also aim to identify the best-performing combinations when these features are assembled into linguistic groups.

5.2 Neural Models

We extend the established usage of transformer-based models in readability assessment (Deutsch et al., 2020; Martinc et al., 2021; Lee et al., 2021) and opt for the BERTurk model[4] for our analysis. We tested multiple learning rates and batch sizes to ascertain the optimal configuration for our task. Specifically, we examined the learning rates of [1e-5, 2e-5, 3e-5, 1e-4] and the batch sizes of [8, 16, 32]. Our final model used AdamW optimizer, linear scheduler with 10% warmup steps, batch size of 8, and learning rate of 3e-5. The sequence lengths of our input documents were all set to 512 tokens. We fine-tune our model for three epochs.

5.3 Hybrid Model

In our study, we experiment with a hybrid model approach that aims to leverage the strengths of both neural and non-neural models in an ensemble learning strategy. The premise behind the hybrid model is based on the observation that while neural models such as BERT have demonstrated robust performance across diverse tasks, they could still benefit from incorporating human-defined linguistic features, which have been key components in traditional non-neural models (Deutsch et al., 2020). Our hybrid model takes a straightforward approach similar to that of Imperial (2021) and Lee et al. (2021). It combines the soft label predictions generated by the neural model with handcrafted features. These features are then used as input to a non-neural (Random Forest) model.

6 Results

We compare the performance of traditional and modern baselines to illustrate the process of arriving at the best-performing model. The process of feature and model selection for the baseline models

[4] https://huggingface.co/dbmdz/bert-base-turkish-uncased

Model	Acc (%)	Rec	Prec	F1
SVM	78.1	78.1	79.0	77.6
RandomF	**85.3**	**85.3**	**85.1**	**85.1**
LogR	83.7	83.6	83.7	83.5
XGBoost	84.1	84.0	84.0	83.7

Table 4: Performance comparison (modern-baseline) of readability models

was carried out based on the results obtained from different combinations.

6.1 Baseline: Feature and Model Evaluation

Linguistic Features	Acc (%)
TRAD	65.7
+ LXSM	76.4
+ SYN	82.5
+ MORPH	83.6
+ DISCO **(ALL)**	**85.3**

Table 5: Incremental contribution of each feature to the RandomF model

Through evaluation of four distinct models, namely Support Vector Machines (SVM), Random Forest (RandomF), Logistic Regression (LogR), and XGBoost, we assessed combinations of five different linguistic groups: traditional (TRAD), lexico-semantic (LXSM), syntactic (SYNX), morphological (MORPH), and discourse (DISCO) features. Table 6 provides a comparative view of these models' performance when trained using the full combination. Among the four models evaluated, the Random Forest model delivered the highest performance with 85.3%. Importantly, all of the linguistic groups used provide orthogonal or distinct information. Table 5 demonstrates how each contributing linguistic group incrementally improves the accuracy of the Random Forest model. Their combined strength ultimately achieves the highest overall accuracy score.

Model	Acc	Prec	Rec	F1
trad-baseline	65.7	67.5	66.8	66.7
modern-baseline	85.3	85.3	85.1	85.1
neural	92.8	93.1	92.6	92.8
hybrid	**96.1**	**96.1**	**95.6**	**95.8**

Table 6: Performance comparison of readability approaches

The varying levels of performance between different approaches is demonstrated in Table 6. The hybrid model, which combines the strengths of both traditional and neural methodologies, outperforms all other models, securing the highest values for accuracy, precision, recall, and F1 score. Following the hybrid model, the neural model performs the best. The neural model (BERT) demonstrates an enhanced ability to capture nuanced characteristics of text readability, exhibiting superior performance to the baseline models without any handcrafted linguistic features. The modern baseline, incorporating five different linguistic subgroups, achieves superior performance compared to the traditional baseline. This highlights the advantage of leveraging an extended set of linguistic features over merely relying on surface-level features typical of traditional readability formulae.

7 Discussion

7.1 Model Interpretation

In order to gain insights into the significance of individual linguistic features within our best-performing model, the RF model, we utilised two well-established model interpretation techniques specifically designed for Random Forest models: Feature Permutation and Mean Decrease in Impurity (MDI) as shown in Figure 1 and 2.

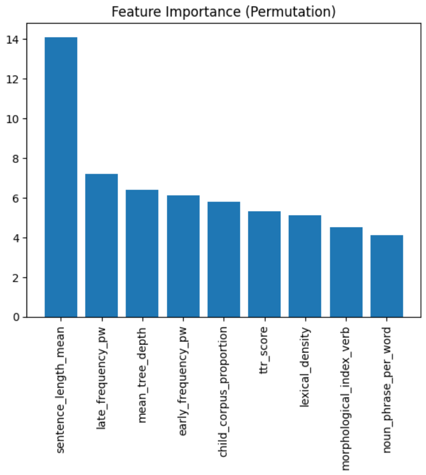


Figure 1: Feature importance by permutation on full model

7.2 Feature Correlation

We also considered model-independent analysis through Spearman correlation to gain additional perspective into the importance of features with respect to readability levels. Table 7 presents the ten features with the highest Spearman correlation

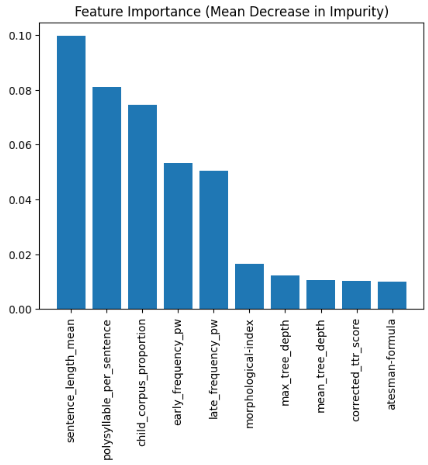


Figure 2: Feature Importance importance by MDI on full model

coefficients highlighting the significance of readability assessment.

Group	Feature	ρ
TRAD	Sentence Length Mean	0.487
TRAD	Polysyllable Count	0.467
LXSM	Child Corpus Proportion	0.433
SYNX	Mean Tree Depth	0.419
LXSM	Lexical Verb Variation	0.403
LXSM	Early Frequency PW	0.385
LXSM	Corrected TTR Score	0.352
LXSM	Lexical Density	0.321
LXSM	Lexical Noun Variation	0.297
SYNX	Noun Phrase Per Word	0.278

Table 7: Top ten features ranked by their Spearman correlation coefficients

7.3 Lingustic Features

The analysis of feature importance consistently highlights the significant role of simple measures such as average sentence length and polysyllable counts. These findings align with previous research, where it has been shown that even compared to more complex feature extraction methods, a simple measure such as sentence length can indirectly capture multiple linguistic aspects of readability. Furthermore, our analysis demonstrates that lexico-semantic features play a prominent role in determining readability. This is evident from the performance improvement observed when including LXSM linguistic feature set in the modern-baseline method. It indicates that while traditional features are indeed valuable, incorporating fine-grained information at the semantic and lexical level can lead to an even better understanding of overall readability. The consistent presence of the syntactic feature "mean tree depth" further supports the relationship between sentence length and syntactic complexity. The correlation between mean tree depth and mean sentence length suggests that the structural complexity captured by syntactic features aligns with the overall complexity of sentences.

8 Conclusion

We introduced a new readability corpus based on popular science magazine articles, providing a valuable resource for future research in Turkish readability assessment. By exploring the effectiveness of linguistic features at different levels, we have demonstrated their superiority over traditional readability formulae and shallow-level features. Our findings emphasise the importance of incorporating fine-grained linguistic features, as they provide more comprehensive insights into the complexity of Turkish texts. We showed the potential of hybrid models that combine fine-grained features with neural models by leveraging the strengths of both linguistic features and state-of-the-art transformers.

Acknowledgements

We would like to pay special thanks to Sefa Kalkan from Charles University for his contributions to the construction of the readability corpus.

References

Elif Ahsen Acar, Deniz Zeyrek, Murathan Kurfalı, and Cem Bozşahin. 2016. A Turkish database for psycholinguistic studies based on frequency, age of acquisition, and imageability. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3600–3606, Portorož, Slovenia. European Language Resources Association (ELRA).

Yakup Akgül. 2019. The accessibility, usability, quality and readability of turkish state and local government websites an exploratory study. *Int. J. Electron. Gov. Res.*, 15(1):62–81.

Yakup Akgül. 2022. Evaluating the performance of websites from a public value, usability, and readability perspectives: a review of turkish national government websites. *Universal Access in the Information Society*, pages 1–16.

Ender Ateşman. 1997. Türkçede okunabilirliğin ölçülmesi. *Dil Dergisi*, 58(71-74).

Burak Bezirci and Asım Egemen Yilmaz. 2010. Metinlerin okunabilirliğinin ölçülmesi üzerine bir yazilim kütüphanesi ve türkçe için yeni bir okunabilirlik ölçütü. *Dokuz Eylül Üniversitesi Mühendislik Fakültesi Fen ve Mühendislik Dergisi*, 12(3):49–62.

Nilay Boztas, Dilek Omur, Sule Ozbılgın, Gözde Altuntas, Ersan Piskin, Sevda Ozkardesler, and Volkan Hanci. 2017. Readability of internet-sourced patient education material related to "labour analgesia". *Medicine*, 96(45).

Nilay Boztaş, Şule Özbilgin, Elvan Öçmen, Gözde Altuntaş, Sevda Özkardeşler, Volkan Hancı, and Ali Günerli. 2014. Evaluating the readibility of informed consent forms available before anaesthesia: a comparative study. *Turkish journal of anaesthesiology and reanimation*, 42(3):140.

Vaclav Brezina and Gabriele Pallotti. 2019. Morphological complexity in written l2 texts. *Second language research*, 35(1):99–119.

Marc Brysbaert and Boris New. 2009. Moving beyond kučera and francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for american english. *Behavior research methods*, 41(4):977–990.

Brian J Bushee, Ian D Gow, and Daniel J Taylor. 2018. Linguistic complexity in firm disclosures: Obfuscation or information? *Journal of Accounting Research*, 56(1):85–121.

Gökhan Çetinkaya. 2010. Türkçe metinlerin okunabilirlik düzeylerinin tanımlanması ve sınıflandırılması. *Ankara University, PhD Thesis*.

Kevyn Collins-Thompson. 2014. Computational assessment of text readability: A survey of current and future research. *ITL-International Journal of Applied Linguistics*, 165(2):97–135.

Michael A Covington and Joe D McFall. 2010. Cutting the gordian knot: The moving-average type–token ratio (mattr). *Journal of quantitative linguistics*, 17(2):94–100.

Edgar Dale and Jeanne S Chall. 1948. A formula for predicting readability: Instructions. *Educational research bulletin*, pages 37–54.

Felice Dell'Orletta, Simonetta Montemagni, and Giulia Venturi. 2011. Read–it: Assessing readability of italian texts with a view to text simplification. In *Proceedings of the second workshop on speech and language processing for assistive technologies*, pages 73–83.

Tovly Deutsch, Masoud Jasbi, and Stuart Shieber. 2020. Linguistic features for readability assessment. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–17, Seattle, WA, USA → 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.

William H DuBay. 2007. *Smart Language: Readers, Readability, and the Grading of Text.* ERIC.

Lijun Feng, Noémie Elhadad, and Matt Huenerfauth. 2009. Cognitively motivated features for readability assessment. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 229–237, Athens, Greece. Association for Computational Linguistics.

Lijun Feng, Martin Jansche, Matt Huenerfauth, and Noémie Elhadad. 2010. A comparison of features for automatic readability assessment. In *23rd International Conference on Computational Linguistics (COLING 2010), Poster Volume*, pages 276–284.

Rudolph Flesch. 1948. A new readability yardstick. *Journal of applied psychology*, 32(3):221.

Thomas François and Cédrick Fairon. 2012. An "AI readability" formula for French as a foreign language. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 466–477, Jeju Island, Korea. Association for Computational Linguistics.

Itziar Gonzalez-Dios, María Jesús Aranzabe, Arantza Díaz de Ilarraza, and Haritz Salaberri. 2014. Simple or complex? assessing the readability of Basque texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 334–344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Robert Gunning. 1969. The fog index after twenty years. *Journal of Business Communication*, 6(2):3–13.

Ahmet Zeki Guven. 2014. Readability of texts in textbooks in teaching turkish to foreigners. *The Anthropologist*, 18(2):513–522.

Julia Hancke, Sowmya Vajjala, and Detmar Meurers. 2012. Readability classification for German using lexical, syntactic, and morphological features. In *Proceedings of COLING 2012*, pages 1063–1080, Mumbai, India. The COLING 2012 Organizing Committee.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, Adriane Boyd, et al. 2020. spacy: Industrial-strength natural language processing in python.

Joseph Marvin Imperial. 2021. Bert embeddings for automatic readability assessment. *arXiv preprint arXiv:2106.07935*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

S Keklik. 2010. Türkçede 0-6 yaş çocuklarına öğretilmesi gereken en sık kullanılan 1200 kelime. *Türkiye Sosyal Araştırmalar Dergisi*, 3(3):1–28.

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

Walter Kintsch and Douglas Vipond. 2014. Reading comprehension and readability in educational practice and psychological theory. *Perspectives on learning and memory*, pages 329–365.

Bruce W. Lee, Yoo Sung Jang, and Jason Lee. 2021. Pushing on text readability assessment: A transformer meets handcrafted linguistic features. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10669–10686, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bruce W Lee and Jason Hyung-Jong Lee. 2023. Traditional readability formulas compared for english. *arXiv preprint arXiv:2301.02975*.

Philip Ley and Tony Florio. 1996. The use of readability formulas in health care. *Psychology, Health & Medicine*, 1(1):7–28.

Xiaofei Lu. 2010. Automatic analysis of syntactic complexity in second language writing. *International journal of corpus linguistics*, 15(4):474–496.

Matej Martinc, Senja Pollak, and Marko Robnik-Šikonja. 2021. Supervised and unsupervised neural approaches to text readability. *Computational Linguistics*, 47(1):141–179.

G Harry Mc Laughlin. 1969. Smog grading-a new readability formula. *Journal of reading*, 12(8):639–646.

Sarah E Petersen and Mari Ostendorf. 2009. A machine learning approach to reading level assessment. *Computer speech & language*, 23(1):89–106.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*.

Tim Smit, Max van Haastrecht, and Marco Spruit. 2021. The effect of countermeasure readability on security intentions. *Journal of Cybersecurity and Privacy*, 1:675–704.

Veysel Sönmez. 2003. Metinlerin eğitselliğini saptamada matematiksel bir yaklaşım). *Marmara University, Master's Thesis*, 10:24–39.

Keziban Tekşan, Üzeyir Süğümlü, and Enes Çinpolat. 2020. Readability of turkish tales. *Journal of Language and Linguistic Studies*, 16(2):978–992.

Sara Tonelli, Ke Tran Manh, and Emanuele Pianta. 2012. Making readability indices readable. In *Proceedings of the First Workshop on Predicting and Improving Text Readability for target reader populations*, pages 40–48, Montréal, Canada. Association for Computational Linguistics.

Tuncay Turkben. 2019. Readability characteristics of texts in middle school turkish textbooks. *Educational Policy Analysis and Strategic Research*, 14(3):80–105.

Sowmya Vajjala. 2022. Trends, limitations and open challenges in automatic readability assessment research. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5366–5377, Marseille, France. European Language Resources Association.

Sowmya Vajjala and Ivana Lučić. 2018. OneStopEnglish corpus: A new corpus for automatic readability assessment and text simplification. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 297–304, New Orleans, Louisiana. Association for Computational Linguistics.

Sowmya Vajjala and Detmar Meurers. 2012. On improving the accuracy of readability classification using insights from second language acquisition. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 163–173, Montréal, Canada. Association for Computational Linguistics.

Lih-Wern Wang, Michael J Miller, Michael R Schmitt, and Frances K Wen. 2013. Assessing readability formula differences with written health information materials: application, results, and recommendations. *Research in Social and Administrative Pharmacy*, 9(5):503–516.

Zarah Weiss, Xiaobin Chen, and Detmar Meurers. 2021. Using broad linguistic complexity modeling for cross-lingual readability assessment. In *Proceedings of the 10th Workshop on NLP for Computer Assisted Language Learning*, pages 38–54, Online. LiU Electronic Press.

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

Comparison of Wav2vec 2.0 Transformer Models for Speaker Change Detection

Zbyněk Zajíc and **Marie Kunešová**
New Technologies for the Information Society and Department of Cybernetics,
Faculty of Applied Sciences, University of West Bohemia, Pilsen, Czech Republic
zzajic@ntis.zcu.cz, mkunes@ntis.zcu.cz

Abstract

The state-of-the-art for various speech tasks is a sequence-to-sequence model based on a self-attention mechanism known as Transformer. The broadly used Wav2vec 2.0 is a self-supervised transformer model pre-trained on large unlabeled datasets and subsequently fine-tuned for a particular task. The data, along with the size of the transformer model, play a crucial role in both these training steps. In this paper, we utilize Wav2vec 2.0 models for finding the speaker change in a speech signal. Our goal is to compare different model sizes with different training datasets to show that data similar to the task domain bring better performance than larger models. The speaker change detection task was tested on four real conversation corpora with consistent top results.

1 Introduction

Speaker change detection (SCD) is the task of finding the point in a conversation where the speaker is changing. It is a basic speech-processing task that is relevant to various speech applications such as speaker diarization (Bullock et al., 2020; Kunešová et al., 2017; Zajíc et al., 2016), automatic speech recognition (Wu et al., 2023), and other tasks related to processing multi-speaker audio (Aronowitz and Zhu, 2020; Zajíc et al., 2018).

Legacy approaches for the SCD task include computing a distance between two sliding windows (Rouvier et al., 2013), detecting differences in pitch (Hogg et al., 2019), or using precomputed features based on i/x-vectors (Aronowitz and Zhu, 2020), Mel-frequency cepstral coefficients (MFCCs) (Hogg et al., 2019), spectrograms (Hrúz and Zajíc, 2017), and combinations of multiple types of features (Su et al., 2022), even including lexical information gained from automated transcripts (Anidjar et al., 2021; Zajíc et al., 2018) or word embeddings (weon Jung et al., 2023) for speaker change detection. Different neural network model architectures have been applied, such as LSTM (Hrúz and Hlaváč, 2018), CNN (Hrúz and Zajíc, 2017), or sequence-level modeling methods (Fan et al., 2022). Nowadays, the transformer network concept uses the attention mechanism of deep learning (Vaswani et al., 2017), which has recently seen great success on a variety of tasks, including but not limited to speech processing (Liu et al., 2021). The main benefit is self-supervised learning on unlabeled data.

In this paper, we investigate the wav2vec 2.0 (Baevski et al., 2020) framework in an end-to-end approach for SCD, first proposed in our previous paper (Kunešová and Zajíc, 2023), where it was shown to achieve state-of-the-art results. The main focus of this paper is to explore the capabilities of different pre-trained wav2vec 2.0 models of various sizes. The results are evaluated on four conversational speech corpora broadly used in the SCD task.

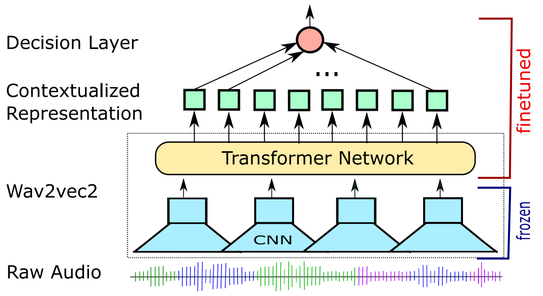


Figure 1: Illustration of the multitask wav2vec 2.0 detector of speaker changes. The model outputs a label for each audio frame (every 20 ms).

2 Wav2vec 2.0 models

Self-supervised audio transformers are known to scale well with the size of pre-training data. Wav2vec 2.0 (hereafter referred to as “wav2vec2”) is a transformer-based self-supervised framework for speech representation, which has been used for a wide range of speech processing tasks, such as automatic speech recognition (Lehečka

Table 1: Pre-trained wav2vec2 models used in this paper.

Model	#Trans.	#Param.	Datasets	Hours	Lang.
wav2vec2-base (Baevski et al., 2020)	12	$\sim 95M$	Librispeech	960	English
wav2vec2-large (Baevski et al., 2020)	24	$\sim 317M$	Librispeech	960	English
wav2vec2-large-xlsr-53 (Conneau et al., 2021)	24	$\sim 317M$	MLS, CV, BABEL	$\sim 56k$	53 lang.
wav2vec2-base-cs-80k-ClTRUS (Lehečka et al., 2022)	12	$\sim 95M$	various	$\sim 80k$	Czech

et al., 2022) and many others (Yang et al., 2021). There is a huge family of these models with different numbers of parameters trained on different datasets. From this zoo, we pick four models[1] for our evaluation: two that were used in (Kunešová and Zajíc, 2023) – the base model `wav2vec2-base` and the large cross-lingual (XLSR) model `wav2vec2-large-xlsr-53`, plus two others. We added the English large model `wav2vec2-large` and, to show the efficiency of models trained on different than clean data, also the Czech model `wav2vec2-base-cs-80k-ClTRUS`, which is trained on data from a greater variety of different domains (Lehečka et al., 2022). Their parameters are summarized in Table 1.

3 Speaker Change Detection (SCD) task

Speaker change in the SCD task is defined as a point in the audio signal where the speaker changes to another speaker, silence, or overlapping speech. The point where a speaker starts to speak after a silence is also a speaker change.

SCD is generally language-independent because language can be seen as one part of the speaker's characteristics. We try to discriminate these speakers from each other (to find their change). On the other hand, the discrepancy in the train and test acoustic domains plays a significant role in the speech representation by the end-to-end model.

The absence of a large quantity of labeled data needed for the deep learning approach forces us to use a self-supervised model as wav2vec2.

3.1 SCD model

As described in our previous paper (Kunešová and Zajíc, 2023), we treat the SCD problem as an audio frame classification task. We use the wav2vec2 model to get a contextual representation of the input signal, with an additional last decision layer as a speaker change detector. The outputs from the transformer are fully connected to the decision layer (one neuron with a linear activation function), which outputs information about the speaker changes in each audio frame every 20 ms, as per the pre-trained wav2vec2 model. Due to the character of the labeling function (see Section 3.2), the model is trained for regression (with mean square error loss) rather than a simple binary classification. The AdamW algorithm was used as an optimizer except for the wav2vec2-large model, where an Adamax provided more stable training behavior.

For the fine-tuning on SCD-labeled data, only the first CNN layer is frozen. For this step, we are using the HuggingFace Transformers (Wolf et al., 2020) library, as in our aforementioned previous paper[2]. The system's architecture is in Figure 1.

Because of the high memory requirements of the wav2vec2 models, the 16 kHz input signal is given in segments of 20 seconds, with a 10-second overlap between segments. Then when the resulting predictions are joined back together for evaluation, we use the middle part of each segment and discard the duplicate 5 s intervals at the edges. This ensures that there is always sufficient context on both sides of a potential speaker change point.

3.2 Reference labels for SCD

Reference labels for the SCD task are based on the annotation files in the Rich Transcription Time Marked (RTTM) format (i.e., the standard annotation format for speaker diarization). Each line in an RTTM file specifies the time interval and speaker ID of one unbroken speaker turn. In our work, we consider the beginnings and ends of all these intervals as speaker change points, with one minor adjustment: during fine-tuning, if two turns of the same speaker have only a small gap (less than one second) between them, we merge the two turns, ignoring the gap. This helps to prevent the model from becoming too sensitive and reporting "speaker changes" even in brief pauses between words.

Additionally, in order to deal with time inaccu-

[1] Downloaded from `https://huggingface.co/facebook/wav2vec2-base`, .../wav2vec2-large, .../wav2vec2-large-xlsr-53 and `https://huggingface.co/fav-kky/wav2vec2-base-cs-80k-ClTRUS`

[2] Our code is available at `https://github.com/mkunes/w2v2_audioFrameClassification`.

Table 2: Our results (%) for SCD task with models fine-tuned either on in-domain data or on an artificial dataset.

Evaluated	**Feature**	**In-domain train data**			**Artificial train data**		
corpus	**model**	Cov	Pur	F1	Cov	Pur	F1
AMI	wav2vec2-base	90.94	90.06	90.50	83.45	81.34	82.38
	wav2vec2-large	91.52	90.31	90.91	80.25	82.77	81.49
	wav2vec2-large-xlsr-53	92.20	90.39	**91.28**	83.45	83.76	**83.61**
	wav2vec2-base-cs-80k-ClTRUS	92.41	89.97	91.18	85.02	79.61	82.22
DH-I	wav2vec2-base	93.74	89.65	91.65	92.93	86.09	89.38
	wav2vec2-large	94.98	89.25	92.03	91.29	87.32	89.26
	wav2vec2-large-xlsr-53	95.56	89.00	**92.16**	89.43	89.79	**89.61**
	wav2vec2-base-cs-80k-ClTRUS	94.61	89.17	91.81	91.04	88.31	89.65
DH-II	wav2vec2-base	92.93	92.09	92.51	95.00	85.90	90.22
	wav2vec2-large	94.75	91.04	92.86	93.67	87.24	90.34
	wav2vec2-large-xlsr-53	95.59	91.19	**93.33**	92.46	89.51	**90.96**
	wav2vec2-base-cs-80k-ClTRUS	94.88	91.45	93.13	95.29	86.75	90.82
CallHome	wav2vec2-base	93.48	92.70	93.09	92.83	86.38	89.49
	wav2vec2-large	92.62	93.36	92.99	89.62	89.40	89.51
	wav2vec2-large-xlsr-53	93.51	93.49	93.50	93.79	88.47	**91.05**
	wav2vec2-base-cs-80k-ClTRUS	94.51	92.54	**93.51**	94.51	84.55	89.25

racies in the human-annotated references, we also use a fuzzy labeling strategy, which we first developed in (Hrúz and Zajíc, 2017): speaker change points are given a reference label with a value of 1, which linearly decreases to zero over an interval of ±0.2 s around each boundary. Audio frames more than 0.2 s away from the nearest speaker change point are labeled as 0.

During evaluation, we detect speaker change points by first finding peaks (local maxima) in the predicted labels and then applying a threshold – peaks above the threshold are considered speaker change points. In this paper, unlike (Kunešová and Zajíc, 2023), we also set a minimum distance between detected peaks as 0.25 s – if there are multiple peaks within 0.25 s, only the highest one is kept (this brings a very minor but consistent improvement in F1-score). However, the fine-tuned “base” and “xlsr-53” models themselves were identical to the previous work. No other post-processing of the model outputs is performed.

4 Datasets

To evaluate the effectiveness of different wav2vec2 models, we tested our system on several widely used English-language conversational speech corpora, which have annotated speaker turns for SCD evaluation.

The tested corpora were the following: AMI Meetings Corpus (**AMI**) (Carletta, 2007), the American English subset of the CallHome (**CallHome**) (Canavan et al., 1997), and the First and Second DIHARD Challenge data (**DH-I**) (Ryant et al., 2018; Bergelson, 2016) and (**DH-II**) (Ryant et al., 2019; Bergelson, 2016).

To also compare the effectiveness of the individual wav2vec2 models on out-of-domain data, we designed a synthetic training dataset in (Kunešová et al., 2019; Kunešová and Zajíc, 2023), made from the LibriSpeech corpus. This way, we can control the speaker change points and also ensure that reference labels are accurate.

5 Results and discussion

Predicted speaker change points were evaluated in terms of audio segmentation, as segment purity (Pur), coverage (Cov), and F1-score, using the Python library `pyannote.metrics`[3] (Bredin, 2017). Purity measures how homogeneous the segments are, and coverage expresses whether each speaker turn is fully contained within one segment. F1-score is the harmonic mean of the two.

Results[4] for individual corpora can be seen in Table 2. We used identical settings for all our models and corpora. We set these values in such a way as to obtain high F1 scores on the AMI development set across all models that were trained or evaluated on AMI – as five training epochs and a threshold of 0.35. The consistency of our tested models is evident from the Coverage vs. Purity graph in Figure 2 for all four corpora.

[3] Downloaded from: `https://pyannote.github.io/`

[4] Unlike our results in (Kunešová and Zajíc, 2023), a minimum distance between peaks (0.25 s) is applied in this study.

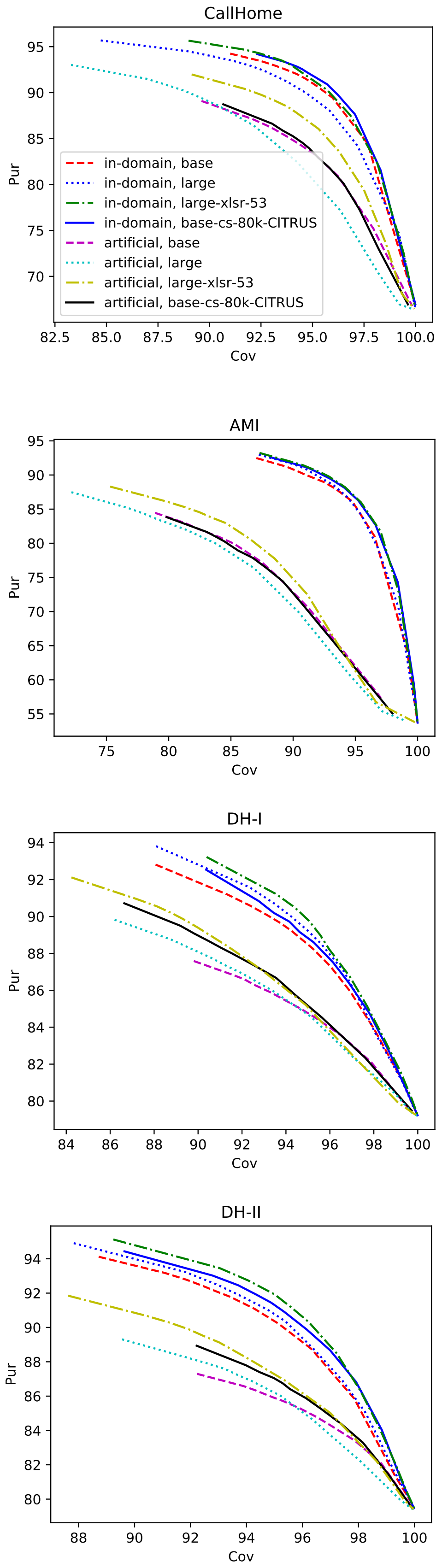


Figure 2: Cov vs. Pur for different thresholds with models fine-tuned on in-domain or artificial data.

Table 3: Previously reported SCD results (%) on different corpora, with models fine-tuned on in-domain data.

Corpus and SCD method	Cov	Pur	F1
AMI (Su et al., 2022)	91.75	85.68	88.61
AMI (Fan et al., 2022)	89.81	83.92	86.76
AMI (Bredin et al., 2020)	84.2	90.4	–
DH-I (Fan et al., 2022)	92.56	86.24	89.29
DH-II (Bredin et al., 2020)	93.7	86.8	–
CallH. (Hrúz and Hlaváč, 2018)	72.57	72.57	–

In comparing the *base* and *large* models, where the number of parameters and the amount of pre-training data are substantially different, the larger models (three times more parameters), especially "xlsr-53", expectedly outperform the base model. The results for the "ClTRUS" model are more interesting. The better-trained "ClTRUS" model with the same architectural size as the base model also consistently brings better results, and is mostly better than the larger models on in-domain data.

The base and large models were trained mainly on clean Librispeech data and are unfamiliar with real wild acoustics conditions in tested data. On the other hand, the "ClTRUS" model saw "wild" data during the pre-training phase, and the fine-tuning on in-domain data can benefit from this. Similarly, the larger "xlsr-53" model, which was trained on more variable data from a few different datasets, also supports this trend.

For a comparison with other systems from different state-of-the-art articles, we present Table 3, showing the best results on the selected corpora we could find in the literature.

6 Conclusion

In this paper, we tested four different wav2vec2 models with an additional decision layer for the SCD task. Wav2vec2 is a relatively complex model with a high computation cost, but we want to use this approach in a transcription system in combination with existing ASR (Lehečka et al., 2022), where the first wav2vec2 layers can be shared. The results of our system with all the tested models surpass all previous results on the same datasets. A comparison of these models shows us the importance of in-domain data not only in fine-tuning phase but also in the self-supervised pre-training phase. According to the results, we believe that richer data for pre-training the models brings more gain than bigger models.

Acknowledgements

This research was supported by the Czech Ministry of Interior, project ROZKAZ (VJ01010108) and by the Czech Ministry of Education, Youth and Sports, Project No. (LM2023062) LINDAT/CLARIAH-CZ. Computational resources provided by the e-INFRA CZ project (ID:90254) were greatly appreciated.

References

Or Haim Anidjar, Itshak Lapidot, Chen Hajaj, Amit Dvir, and Issachar Gilad. 2021. Hybrid speech and text analysis methods for speaker change detection. *IEEE/ACM Transactions on Audio Speech and Language Processing*, 29:2324–2338.

Hagai Aronowitz and Weizhong Zhu. 2020. Context and uncertainty modeling for online speaker change detection. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 8379–8383.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

Elika Bergelson. 2016. Bergelson Seedlings HomeBank Corpus.

Hervé Bredin. 2017. pyannote.metrics: A toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems. In *Proc. Interspeech*, pages 3587–3591.

Hervé Bredin et al. 2020. pyannote.audio: Neural building blocks for speaker diarization. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 7124–7128.

Latané Bullock, Hervé Bredin, and Leibny Paola Garcia-Perera. 2020. Overlap-aware diarization: Resegmentation using neural end-to-end overlapped speech detection. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 7114–7118.

Alexandra Canavan, David Graff, and George Zipperlen. 1997. CALLHOME American English Speech, LDC97S42. In *LDC Catalog*. Linguistic Data Consortium.

Jean Carletta. 2007. Unleashing the killer corpus: experiences in creating the multi-everything AMI Meeting Corpus. *Language Resources and Evaluation*, 41(2):181–190.

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. 2021. Unsupervised cross-lingual representation learning for speech recognition. In *Proc. Interspeech*, pages 2426–2430.

Zhiyun Fan, Linhao Dong, Meng Cai, Zejun Ma, and Bo Xu. 2022. Sequence-level speaker change detection with difference-based continuous integrate-and-fire. *IEEE Signal Processing Letters*, 29:1551–1554.

Aidan O.T. Hogg, Christine Evers, and Patrick A. Naylor. 2019. Speaker change detection using fundamental frequency with application to multi-talker segmentation. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 5826–5830.

Marek Hrúz and Miroslav Hlaváč. 2018. LSTM neural network for speaker change detection in telephone conversations. *Speech and Computer. SPECOM 2018. Lecture Notes in Computer Science*, 11096:226–233.

Marek Hrúz and Zbyněk Zajíc. 2017. Convolutional neural network for speaker change detection in telephone speaker diarization system. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 4945–4949.

Marie Kunešová, Marek Hrúz, Zbyněk Zajíc, and Vlasta Radová. 2019. Detection of overlapping speech for the purposes of speaker diarization. *Speech and Computer. SPECOM 2019. Lecture Notes in Computer Science*, 11658:247–257.

Marie Kunešová and Zbyněk Zajíc. 2023. Multitask detection of speaker changes, overlapping speech and voice activity using wav2vec 2.0. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1–5.

Marie Kunešová, Zbyněk Zajíc, and Vlasta Radová. 2017. Experiments with segmentation in an online speaker diarization system. *Text, Speech, and Dialogue. TSD 2017. Lecture Notes in Computer Science*, 10415:429–437.

Jan Lehečka, Jan Švec, Aleš Pražák, and Josef V. Psutka. 2022. Exploring capabilities of monolingual audio transformers using large datasets in automatic speech recognition of Czech. In *Proc. Interspeech*, pages 1831–1835.

Andy T. Liu, Shang-Wen Wen Li, and Hung-yi Yi Lee. 2021. TERA: Self-supervised learning of transformer encoder representation for speech. *IEEE/ACM Transactions on Audio Speech and Language Processing*, 29:2351–2366.

Mickael Rouvier, Grégor Dupuy, Paul Gay, Elie Khoury, Teva Merlin, and Sylvain Meignier. 2013. An open-source state-of-the-art toolbox for broadcast news diarization. In *Proc. Interspeech*, pages 1477–1481.

Neville Ryant et al. 2018. First DIHARD Challenge evaluation plan. Technical report, Linguistic Data Consortium.

Neville Ryant et al. 2019. The Second DIHARD Diarization Challenge: Dataset, task, and baselines. In *Proc. Interspeech*, pages 978–982.

Hang Su et al. 2022. A multitask learning framework for speaker change detection with content information from unsupervised speech decomposition. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 8087–8091.

Ashish Vaswani et al. 2017. Attention is all you need. In *Proc. 31st International Conference on Neural Information Processing Systems (NIPS'17)*, pages 5998–6008.

Jee weon Jung et al. 2023. Encoder-decoder multimodal speaker change detection. In *Proc. Interspeech*, pages 5311–5315.

Thomas Wolf 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.

Jian Wu, Zhuo Chen, Min Hu, Xiong Xiao, and Jinyu Li. 2023. Speaker change detection for transformer transducer ASR. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1–5.

Shu-wen Yang et al. 2021. SUPERB: Speech processing Universal PERformance Benchmark. In *Proc. Interspeech*, pages 1194–1198.

Zbyněk Zajíc, Marie Kunešová, and Vlasta Radová. 2016. Investigation of segmentation in i-vector based speaker diarization of telephone speech. *Speech and Computer. SPECOM 2016. Lecture Notes in Computer Science*, 9811:411–418.

Zbyněk Zajíc, Daniel Soutner, Marek Hrúz, Luděk Müller, and Vlasta Radová. 2018. Recurrent neural network based speaker change detection from text transcription applied in telephone speaker diarization system. *Text, Speech, and Dialogue. TSD 2018. Lecture Notes in Computer Science*, 11107:342–350.

Zbyněk Zajíc et al. 2018. First insight into the processing of the language consulting center data. *Speech and Computer. SPECOM 2018. Lecture Notes in Computer Science*, 11096:778–787.

Typological classification of European Portuguese fricatives: a cross-language forced alignment and pronunciation variants study

Anisia Popescu and **Lori Lamel** and **Ioana Vasilescu**
Laboratoire Interdisciplinaire des Sciences du Numérique (LISN)
Université Paris Saclay
anisia.popescu@universite-paris-saclay.fr

Abstract

Devoicing in European Portuguese fricatives is an extensive phenomenon, especially when compared to other languages. Small scale acoustic studies have shown that devoicing rates and voicing profiles of fricatives are more similar to those of Germanic languages, setting European Portuguese (EP) apart within the Romance language family. The present study tests whether voicing in EP fricatives diverges from its sister languages by using empirically motivated combinations of different languages (EP, Italian, German) acoustic phone models on large EP corpora, allowing an ASR system to choose the best fitting one when force aligning the data. Results confirm that voicing in EP fricatives is more similar to German, suggesting EP voicing patterns are shifting away from classic voicing systems known for Romance languages.

1 Introduction

Romance languages are generally known as "true voicing languages" - implementing the voice-voiceless contrast through the use of the [voice] feature (Lisker and Abramson, 1964), opposing prevoiced with unaspirated voiceless obstruents. "Aspirating languages", such as most of the Germanic languages, make use of the [spread glottis] feature (Jansen, 2004), contrasting unaspirated, phonetically voiceless obstruents with long-lag aspirated obstruents. There are however exceptions such as Dutch (van Alphen and Smits, 2004) or, more recently, European Portuguese (Pape and Jesus, 2011, 2015). More specifically, in the latter case, both small scale acoustic studies (Jesus and Shadle, 2003; Pape and Jesus, 2011) and large scale corpus-based studies (Wu et al., 2022; Hutin et al., 2022) have found higher rates of devoicing of phonemically voiced obstruents in EP than in other Romance languages. Furthermore evidence from voicing profiles shows that while Italian and Spanish voicing probability remains high (close to 1) throughout the obstruent, in EP and German there is a decrease in voicing probability starting with 30% of the obstruent. (Pape and Jesus, 2015; Shih and Möbius, 1999, 1998).

The present study tests these patterns on a much larger scale (100+ hours of speech) via forced alignment of the speech with the orthographic transcription. EP speech data is aligned using parallel multiple-language acoustic models and pronunciation variants for fricatives to answer the following theoretical research question:

- Does EP fricative voicing show consistency within the Romance languages family, or does it take a different path, more similar to languages that are both genetically and geographically different?

To answer this question we chose one Germanic - German - and one Romance - Italian - language. The choice of languages mirrors the set of languages tested in the original acoustic study (Pape and Jesus, 2015). Based on the chosen language set we can now fine-tune our experimental research question to:

- Is voicing in EP fricatives more similar to German than to Italian ?

2 Methods

To answer this experimental research question we analyzed an EP corpus consisting of 114 hours of mostly standard dialectal broadcast news speech from TV and radio shows. Multiple sources were used for acquiring the data: LDC, ELRA and international projects. The phone level segmentation was generated using a Portuguese acoustic model, estimated using language-specific annotated (manual transcription) training data and pronunciation dictionaries. The output is a sequence of phone segments with labels selected by aligning the reference transcriptions via a language specific dictionary.

To test whether voicing in EP fricatives is more similar to German than Italian, two additional sets of fricative phone models, one for German and one for Italian, were added in parallel to the original Portuguese one. The phone models for all other phonemes are kept in their original Portuguese form. For each language the acoustic models were all trained on roughly 100 hours of transcribed broadcast news data (Portuguese: 1.1 million word tokens, 46k word types; Italian: 1.8 million word tokens, 58.8k word types; German: 1.8 million word tokens, 90k word types). All three (EP, Italian and German) acoustic models are speaker-, context- and word-position-independent monophone models. Each phone model is a 3-state left-to-right continuous density hidden HMM with Gaussian mixtures with up to 32 Gaussians per state (silences are modeled by a single state with 256 Gaussians). Each acoustic model used the same acoustic parameterization (cepstral - PLP (Hermansky, 1990) and pitch (F0) features), similar to (Lamel et al., 2011). Figure 1 illustrates the speech modeling and alignment process. By using different combinations of language acoustic models on EP speech data we force the recognition system to choose the best fitting phone model (be it the original EP, or an Italian or German one) for each individual phonemically voiced fricative in the corpus (illustrated in Figure 2). The set of voiced fricatives in the corpus consisted of a total of 37.563 coronal /z/ and 36.354 labiodental /v/. The postalveolar /ʒ/ was left out of the study since it is not included in the Italian phoneme inventory and it appears only in loanwords in German.

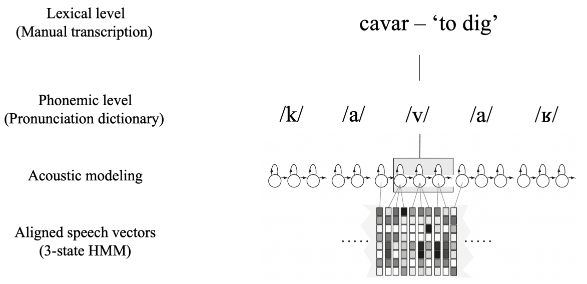


Figure 1: Illustration of the speech modeling and alignment process for the Portuguese word *cavar* 'to dig'

Two different combinations of the three acoustic models were tested: (1) a three way choice of acoustic models between Portuguese, Italian or German, and (2) a two way choice of acoustic models between Italian or German (Portuguese fricative phone models were no longer available to the ASR system). Each combination will be described in a different section. If voicing in EP fricatives is indeed more similar to German, as attested by previous acoustic studies, we would expect the system to choose the German fricative phone models to a higher degree than it does the Italian fricative phone models. If, however the opposite stands (i.e., EP voicing does not behave similarly to a Germanic language, but is still related to its sister language Italian), we would expect the system to prefer the Italian fricative phone models.

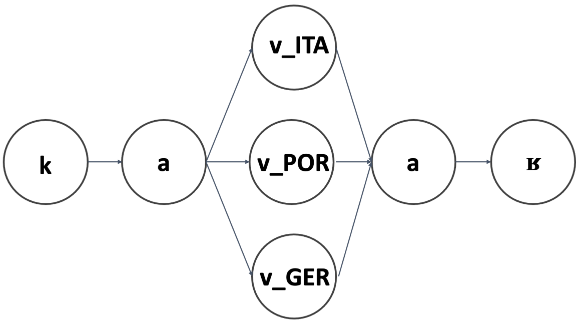


Figure 2: Combination of three acoustic models (Italian, EP and German) for the fricative /v/ in the Portuguese word *cavar* 'to dig'

We ran a third experiment which involved using one acoustic model at a time (not in parallel) with the addition of pronunciation variants. The language specific (European Portuguese) dictionary was enriched with pronunciation variants for fricative voicing. For example the Portuguese word /vinho/ - 'wine' had two possible pronunciations: the original [viɲo] and a devoiced variant [fiɲo]. The procedure is similar to that described for English and French in (Lamel and Adda, 1996; Adda-Decker and Lamel, 1999). The system then had to choose which phone model (phonetically voiced or phonetically voiceless) best fitted the phonemically voiced fricative in the data. This experiment will be described in section 5.

3 Experiment 1: Three-way choice of acoustic models - European Portuguese, Italian and German

In this first experiment, for each fricative /v,z / the system was presented with three phone models in parallel, the original EP phone model completed with the German and Italian ones. The system then had to choose which of the three phone models for the phonemically voiced /v,z/ best fitted the acoustic realization of the fricatives in European Portuguese. Figure 3 shows the percentages of selected phone models per language as a function of place of articulation (labiodental /v/, coronal /z/).

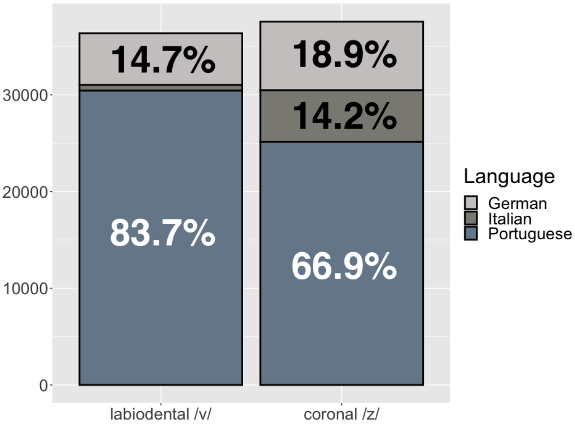


Figure 3: Percentages of phone occurrences aligned with either the original Portuguese, the Italian or the German acoustic model for the labiodental /v/ (left) and the coronal /z/ (right)

As expected, the original Portuguese phone model was markedly preferred (83.7% of cases for the labiodental /v/ and 66.9% of cases for the coronal /z/). For the rest of the cases the system preferred either the German or the Italian phone models. For both the labiodental /v/ and the coronal /z/ the German models were preferred. There is an effect of place of articulation with more mismatches (i.e., the original Portuguese model is less preferred) in the case of the more back fricative (coronal /z/).

4 Experiment 2: Two-way choice of acoustic models - Italian and German

In the second experiment the original EP phone models for fricatives was no longer an option, forcing the ASR system to choose between either an Italian or a German fricative phone model. Figure 4 shows the counts and percentages of phone occurrences aligned with the Italian or the German phone model as a function of place of articulation.

Results mirror those of experiment one, suggesting the German acoustic models seem to be preferred in 89.5% of cases for the labiodental /v/ and 61.7% of cases for the coronal /z/ over the Italian acoustic models. Similar to experiment 1 there is an effect of place of articulation with Italian acoustic models being chosen to a higher degree in the case of coronal /z/ as compared to the labiodental /v/.

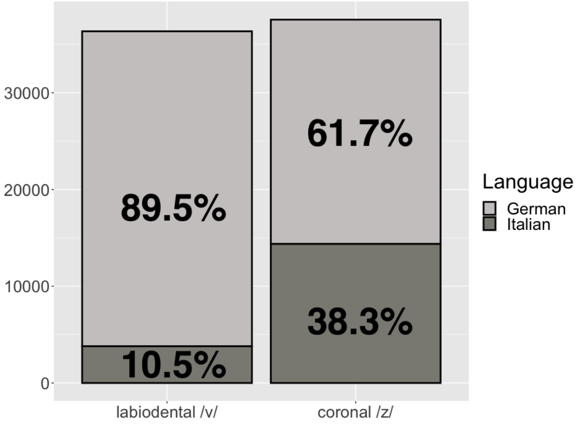


Figure 4: Percentages of phone occurrences aligned with either the Italian or the German acoustic model for the labiodental /v/ (left) and the coronal /z/ (right)

5 Experiment 3: Italian and German with pronunciation variants

In this third experiment, the Portuguese language dictionary was enriched with pronunciation variants for fricative voicing (voiced fricatives /v,z/ could be produced either as phonetically voiced [v,z] or voiceless [f,s]) allowing the system to choose the best phone model (voiced or voiceless) for each phonemically voiced fricative in the data. Two separate alignments were run using either the Italian or the German [v,z - f,s] phone models. For example, when using the Italian acoustic model for fricatives on the data, if a Portuguese phonemically voiced fricative /v/ better matched the Italian phone model [v] the output would be [v]. If however the acoustic realization of the Portuguese [v] better matched the Italian [f] phone model, the output would be the Italian [f]. The same procedure was applied using the German fricative phone models. This experiment differs from the first two, in that it allows us to test the similarity/difference between EP and Italian/German from a different angle. Based on previous acoustic studies, we know that voicing profiles differ based on language: while Italian voicing probability remains high (close to 1) throughout the fricative, the EP and German voicing probability decreases starting with 30% of the fricative (Pape and Jesus, 2015). This suggests that both EP and German exhibit partial devoicing during the fricative, whilst Italian does not. If this is indeed the case we would expect to find higher percentages of voiceless variants when us-

ing the Italian acoustic model (i.e., Italian voiceless fricative models better match the partially devoiced phonetically voiced EP fricative).
Figure 5 shows the percentages of phonetically voiceless variants (greyer shades) identified when using the Italian and German voiced-voiceless acoustic models. White shades correspond to the phonetically voiced variants identified by the system. Results yet again confirm the higher degree of similarity between EP and German fricatives. As predicted when aligning the data with the Italian fricative phone models, the voiceless variants are preferred at higher rates than in the case of the German alignment: for the phonemically voiced Portuguese /v/ the Italian [f] models are preferred in 89.8% of cases compared to only 72.2% cases of German [f]. For the Portuguese phonemically voiced /z/ the Italian [s] models are preferred in 72.2% of cases compared to only 29.9% cases of German [s]. All the attested differences are statistically significant.

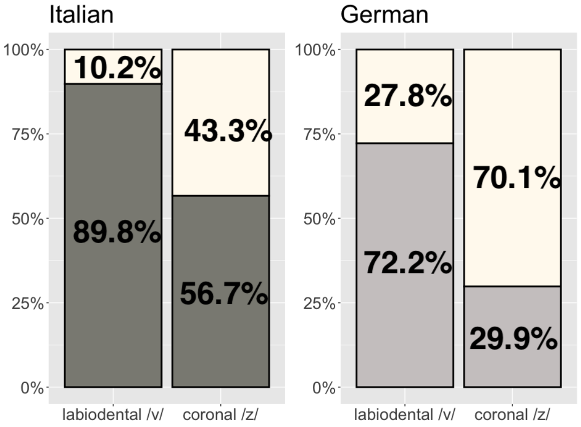


Figure 5: Percentages of phone occurrences aligned with either the voiceless (grey shades) or the voiced (white shade) for the labiodental /v/ (left columns) and coronal /z/ (right columns) per language (Italian on the left and German on the right)

Limitations

The goal of the present study was to (in)validate typological classification results derived from small scale acoustic studies on large scale corpus data. The proposed methodology (i.e., using different combinations of trained acoustic phone models) does not permit a direct replication: while the acoustic studies relied on Praat's (Boersma and Weenink, 2019) autocorrelation (AC) pitch extraction algorithm, the present study relies on the acoustic models of the systems, which include multiple acoustic features. An acoustic analysis pinpointing the most relevant acoustic features is needed. A second limitation of the current study is the non-inclusion of several acoustic correlates of voicing in the analysis. It is known that adjacent segments, position in the word/syllable and stress have a significant effect on devoicing rates (Pape et al., 2003; Bybee and Easterday, 2019; Hutin et al., 2022). The phonological/phonetic context is all the more pertinent given the use of German as a comparison language, whose acoustic correlates for voicing are dependent on word position (word medial vs. initial vs. final) and stress (Jessen, 1998; Fuchs, 2005). Positional effects in the case of fricatives are more reduced in the case of EP since the only licensed consonants in coda position are /l/, /r/ and /ʃ/ and word initial /v,z/ are rare (in our data /v/: 1671 tokens and /z/: 122 tokens) and found mainly in loanwords. Stress on the other hand is relevant: EP, like German, and unlike Italian is said to be stress-timed (Cruz-Ferreira, 1999) or partially stressed and partially syllable-timed (Frota and Vigário, 2006). The corpora is not annotated for prosodic information which limits our analysis. Another well known correlate of voicing that has only indirectly been included in the analysis (via the trained acoustic models) is segmental duration. A more detailed analysis including phonological context and duration information is needed to better explain the patterns.

6 Conclusion

The present study tested whether voicing in European Portuguese fricatives is more similar to Italian, a closely related Romance language, or to German, a more distantly related language, on large scale corpora using ASR acoustic modeling and pronunciation variants. By allowing the system to choose a preferred model when making the alignment we replicated results from small scale acoustic studies that showed EP tends to diverge from other Romance languages when it comes to fricative voicing. The results also show that the effect seems to be modulated by place of articulation, with the more posterior fricative (the coronal /z/) behaving differently than the more anterior labiodental /v/.
Results support the use of speech technology methodologies to replicate and test phonological hypotheses on large amounts of data (Yuan and Liberman, 2011; Ryant et al., 2013).

Acknowledgements

This work was partially supported by the French National Research Agency (ANR) as part of the DIPVAR project (ANR-21-CE38-0019).

References

Martine Adda-Decker and Lori Lamel. 1999. Pronunciation variants across system configuration, language and speaking style. *Speech Communication*, 29(2-4):83–98.

Paul Boersma and David Weenink. 2019. Praat: doing phonetics by computer. *Computer program.*

Joan Bybee and Shelece Easterday. 2019. Consonant strenghtening: a crosslinguistic survey and articulatory proposal. *Linguistic Typology*, 2(23):263–302.

Madalena Cruz-Ferreira. 1999. *Portuguese (European)*. Cambridge University Press.

Sonia Frota and Marina Vigário. 2006. On the correlates of rhythmic distinctions: The European/Brazilian Portuguese case. *Probus*, 13(2):247–275.

Susanne Fuchs. 2005. *Articulatory correlates of the voicing contrast in alveolar obstruent production in German*. Phd thesis, ZAS, Berlin, Germany.

Hynek Hermansky. 1990. Perceptual linear prediction (plp) analysis for speech. *Journal of the Acoustical Society of America*, 87.

Mathilde Hutin, Martine Adda-Decker, Lori Lamel, and Ioana Vasilescu. 2022. When phonetics meets morphology: Intervocalic voicing within and across words in Romance languages. In *Proceedings of the 23rd Annual Conference of the International Speech Communication Association*, pages 3438–3442.

Wouter Jansen. 2004. *Laryngeal contrast and phonetic voicing: a laboratory phonology approach to English, Hungarian and Dutch*. Dissertation, University of Groningen.

Michael Jessen. 1998. *Phonetics and Phonology of Tense and Lax Obstruents in German*. John Benjamins Publishing Company.

Luis M. Jesus and Christine H. Shadle. 2003. Devoicing measures of european portuguese fricatives. In *Proceedings of the International Conference on Computational Processing of the Portuguese Language*, volume 2721.

Lori Lamel and Gilles Adda. 1996. On designing pronunciation lexicons for large vocabulary continuous speech recognition. In *Proc. 4th International Conference on Spoken Language Processing*, Philadelphia, USA.

Lori Lamel, Sandrine Courcinous, Julien Despres, Jean-Luc Gauvain, Yvan Josse, Kevin Kilgour, Florian Kraft, Viet-Bac Le, Hermann Ney, Markus Nußbaum-Thom, Ilya Oparin, Tim Schlippe, Ralf Schlüter, Tanja Schultz, Thiago Fraga da Silva, Sebastian Stüker, Martin Sundermeyer, Bianca Vieru, Ngoc Thang Vu, Alexander Waibel, and Cécile Woehrling. 2011. Speech recognition for machine translation in quaero. In *Proceedings of the 8th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 121–128, San Francisco, California.

Leigh Lisker and Arthur S. Abramson. 1964. A cross-language study of voicing in initial stops: Acoustical measurements. *Word*, (20):384–422.

Daniel Pape and Luis M.T. Jesus. 2011. Devoicing of phonologically voiced obstruents: is European Portuguese different from other Romance languages. In *Proceedings of the 23rd International Conference of Phonetic Sciences*, pages 1566–1569.

Daniel Pape and Luis MT Jesus. 2015. Stop and fricative devoicing in European Portuguese, Italian and German. *Language and Speech*, 58(2):224–245.

Daniel Pape, Christine Mooshamer, Phil Hoole, and Susanne Fuchs. 2003. Devoicing of word-initial stops: a consequence of the following vowel? In *Proceedings of the 6th International Seminar on Speech Production*, pages 207–212, Sydney, Australia.

Neville Ryant, Jiahong Yuad, and Mark Liberman. 2013. Automating phonetic measurement: The case of voice onset time. In *In Proceedings of Meetings on Acoustics ICA2013*, Montreal, Canada.

Chilin Shih and Bernd Möbius. 1998. Contextual effects on voicing profiles of German and Mandarin consonants. In *Proceedings of the 14th International Congress of Phonetic Sciences*, pages 989–992.

Chilin Shih and Bernd Möbius. 1999. Contextual effects on consonantal voicing profiles: A cross-linguistic study. In *Proceedings of the 14th International Congress of Phonetic Sciences*, pages 989–992, San Francisco, US.

Petra M. van Alphen and Roel Smits. 2004. Acoustical and perceptual analysis of the voicing distinction in Dutch initial plosives: the role of prevoicing. *Journal of Phonetics*, 32(4):455–491.

Yaru Wu, Mathilde Hutin, Ioana Vasilescu, Lori Lamel, and Martine Adda-Decker. 2022. Extracting linguistic knowledge from speech: A study of stop realization in 5 Romance languages. In *Proceedings of the International Conference on Language Resources and Evaluation*, pages 3257–3263.

Jiahong Yuan and Mark Liberman. 2011. /l/ variation in american english: A corpus approach. *Journal of Speech Sciences*, (2):35–46.

Methods for Phonetic Scraping of Youtube Videos

Adrien Méli [1], Steven Coats [2], Nicolas Ballier [1] [3]
[1] CLILLAC-ARP / [3] LLF / Université Paris Cité, F-75013 Paris, France
[2] Faculty of Humanities, University of Oulu, FI-90014, Finland
adrienmeli@gmail.com, nicolas.ballier@u-paris.fr, steven.coats@oulu.fi

Abstract

This paper discusses two pipelines for the automatic collection of automatic speech recognition (ASR) transcripts and audio content from YouTube videos and subsequent phonetic analysis: PEASYV (Phonetic Extraction and Alignment of Subtitled YouTube Videos) and YTPP (YouTube Phonetics Pipeline). The pipelines differ somewhat in terms of processing steps as well as the tools used for forced alignment, but produce comparable results. The two pipelines may be useful for large-scale collection of acoustic data for phonetic analysis.

1 Introduction

Widespread availability of high-quality audio and rapid advances in the quality of ASR transcripts have opened new doors for data collection in phonetics. This paper presents two systems designed to collect transcript and audio data from YouTube for the purposes of phonetic forced alignment and analysis: PEASYV (Phonetic Extraction and Alignment of Subtitled YouTube Videos) and YTPP (YouTube Phonetics Pipeline). The pipelines make use of open-source libraries collect data from YouTube, align the transcripts with the audio tracks, and analyze the acoustic data therein. While both pipelines make use of yt-dlp for data collection, PEASYV aligns audio with text by means of the Penn Forced Aligner (p2f) and SPPAS (SPeech Phonetization Alignment and Syllabification, Bigi 2012), and YTPP uses the Montreal Forced Aligner (McAuliffe et al., 2017a). For Acoustic analysis, for example of F1 and F2 formant values, both pipelines ultimately use Praat (Boersma and Weenink, 2023). YTPP is Python-based and its code is available (see Section 4 below).

The rest of the paper is structured as follows. Section 2 discusses a few papers in which forced aligners are compared. Section 3 provides details on PEASYV, and Section 4 introduces YTPP. In Sections 3 and 4, as proof of concept, we demonstrate analyses of an example YouTube video using the two pipelines. Section 5 provides a brief summary and future outlook, including caveats that may be relevant for the automatic harvesting of phonetic data from YouTube and other platforms.

2 Forced aligner comparisons

Forced alignment of speech, or the exact matching of an audio transcript with an audio file, is a necessary prerequisite for the phonetic analysis of acoustic segments such as phrases, words, or phones. A number of software tools have been developed for forced alignment, for example the Munich Automatic Segmentation System (MAUS), which has a web-based implementation (Kisler et al., 2017). Many are based on HTK, the Hidden Markov Model Toolkit (Young et al., 2006), or Kaldi (Povey et al., 2011). The Penn Forced Aligner is based on HTK, while the Montreal Forced Aligner builds on Kaldi. The SPPAS aligner is derived from Julius (Lee and Kawahara, 2019).

MacKenzie and Turton (2020) compared alignments for British English speech produced by composite tools that build upon HTK and Kaldi: They found that while both underlying algorithms produce acceptable alignments, the Montreal Forced Aligner (built upon Kaldi) performed somewhat better than the Penn Forced Aligner (built upon HTK). Similarly, Gonzalez et al. (2020) compared several aligners for Australian speech, finding them to be suitable even when using default models trained on American English. They found a Kaldi-based aligner to be slightly better than HTK-based aligners.

3 PEASYV: Phonetic Extraction and Alignment of Subtitled YouTube Videos

PEASYV is a modular tool for phonetic analysis of YouTube content. The workflow of the tool is automatically managed by shell scripts providing the sequence of commands described in Figure 1. Subtitled videos are scraped by yt-dlp. The downloaded video is then converted to a wav file using ffmpeg, and the subtitles file is converted to a preliminary TextGrid using praat (Boersma and Weenink, 2023). The time stamps from the subtitles serve as boundaries for the TextGrid, and the created intervals are labeled with the subtitles themselves. The sound file and the TextGrid are then split into short files extracted from the intervals. These short sound files, usually lasting under three seconds, are then fed into two forced alignment tools, SPPAS (Bigi, 2012) and the Penn Phonetics Lab Forced Aligner (P2FA, p2f). Both aligners use the Carnegie Mellon University dictionary (CMU, Weide 1994) for grapheme to phoneme correspondences[1]. This procedure contains potential cascading alignment errors and increases accuracy. The resulting short TextGrids are then concatenated back into the main TextGrid, and syllabic tiers, one for each aligner, are created following the syllabification of the *Longman Pronunciation Dictionary* (LPD, Wells 2008). Extra steps are taken regarding prosodic annotation but their description falls beyond the scope of this article (*cf.* Méli and Ballier 2023 for further details). The resulting main TextGrid features segmental, syllabic, and lexical tiers for both aligners, and a Momel (Hirst and Espesser, 1993; Hirst, 2007) and INTSINT tier for SPPAS;[2] two "matching" tiers have also been added (see below). Finally, vocalic data is collected in separate csv spreadsheets, one for each aligner.

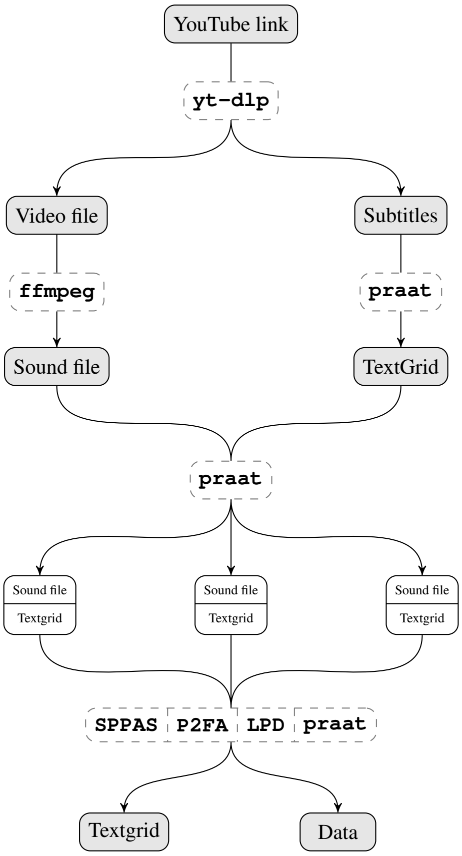


Figure 1: The PEASYV workflow.

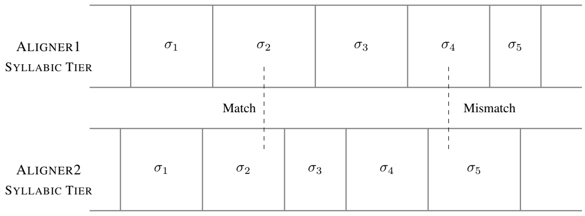


Figure 2: Schematic representations of a "MATCH" (left) and "MISMATCH" (right) case on a PRAAT TextGrid.

Because PEASYV uses two aligners based on two different speech recognition engines (Julius and HTK), assessing the degree of agreement of the generated alignments may arguably provide some insight into their reliability, if not their accuracy. This can be done by comparing local discrepancies and measuring the frequencies of these discrepancies. One way to do this is by flagging vocalic datapoints on a given aligner according to whether the other aligner matches these datapoints. PEASYV implements one such system, and its characteristics are represented in Figure 2. PEASYV uses the LPD-based syllabic tiers as references. The midpoint of σ_1's duration on Aligner1's tier, marked by a vertical dashed line, falls within the boundaries of σ_1's duration on Aligner2's tier. When collecting the phonetic data (*e.g.* formants) corresponding to σ_1 as aligned by Aligner1, the vowel will be marked as "matching". Conversely, σ_4's midpoint on Aligner1's tier, marked by the dashed line on the right, falls outside σ_4's interval on Aligner2's tier: it will therefore be marked as "mismatching".

This experimental feature makes it possible to filter out potential alignments errors and obtain more reliable measurements, especially for sizeable datasets. In contrast with other forced aligners, PEASYV also enables direct comparisons, on the same TextGrid, of two aligners, and provides syl-

[1] The transcriptions of the CMU are however different: SPPAS uses a version of SAMPA, P2FA ARPAbet.

[2] "Momel" stands for "Modelling melody", "INTSINT" for "INternational Transcription System for INTonation".

labic tiers for future analyses.

3.1 Results

Table 1 presents the total number of vowels aligned by SPPAS and P2FA respectively. 27.4% of all 2661 SPPAS-aligned vowels appear in syllables whose mid-temporal values are not included within the corresponding P2FA-generated intervals (*i.e.* they are flagged as "mismatching"). 30.8% of the 2743 P2FA-aligned vowels are "mismatching".

	SPPAS	**P2FA**
Vowels:	2661	2743
– in matching syllables:	1933	1899
– in mismatching syllables:	728	844

Table 1: Per-aligner counts of vowels.

The PEASYV-generated vocalic spaces for monophthongs in a video chosen for test purposes with the identifier _P7_69FeqnU are represented in Figure 3. The formant values of each monophtong are plotted in the F1/F2 space. The ellipses encompass the values within one standard deviation of all the measurements for each monophthong. The label of each monophthong is located at the center of each value (*i.e.* the mean F1/F2 values of the vowel's measurements), and the number next to it gives the number of tokens detected for that vowel. The top row (*i.e.* Figures 3a and 3b) features all monophthongs, while the bottom row (*i.e.* Figures 3c and 3d) only features matching monophthongs (*cf.* previous section and Figure 2).

3.2 Discussion and Prospects

Cursory visual inspection of Figure 3 shows that restricting the data to matching cases yields ellipses which are more clearly defined and less overlapping than using all vowels, regardless of whether their alignment on a given aligner matches that of the other aligner. This is particularly clear with SPPAS-aligned mid front vowels and back vowels. One striking characteristic is the great variation that formant measurements for vowel /uː/ undergo compared to its token count. We contend that the matching procedure may be a simple way to filter out outliers and improve the quality of the extracted data, although no ground truth alignment has been prepared. Of course, the quality of PEASYV-generated data is highly dependent on the original quality of the subtitles. Future research will have to establish whether transcriptions based on automatic speech recognition systems such as Whisper yield more reliable data.

PEASYV is meant to be deployed on a website[3] where links to subtitled videos can be uploaded and generated TextGrids can be downloaded. The source code may also be made partially available for deployment on Linux servers. PEASYV will hopefully be useful to study less common varieties of English. Corpora of Nigerian and Ugandan English are under way.

4 YTPP: YouTube Phonetics Pipeline

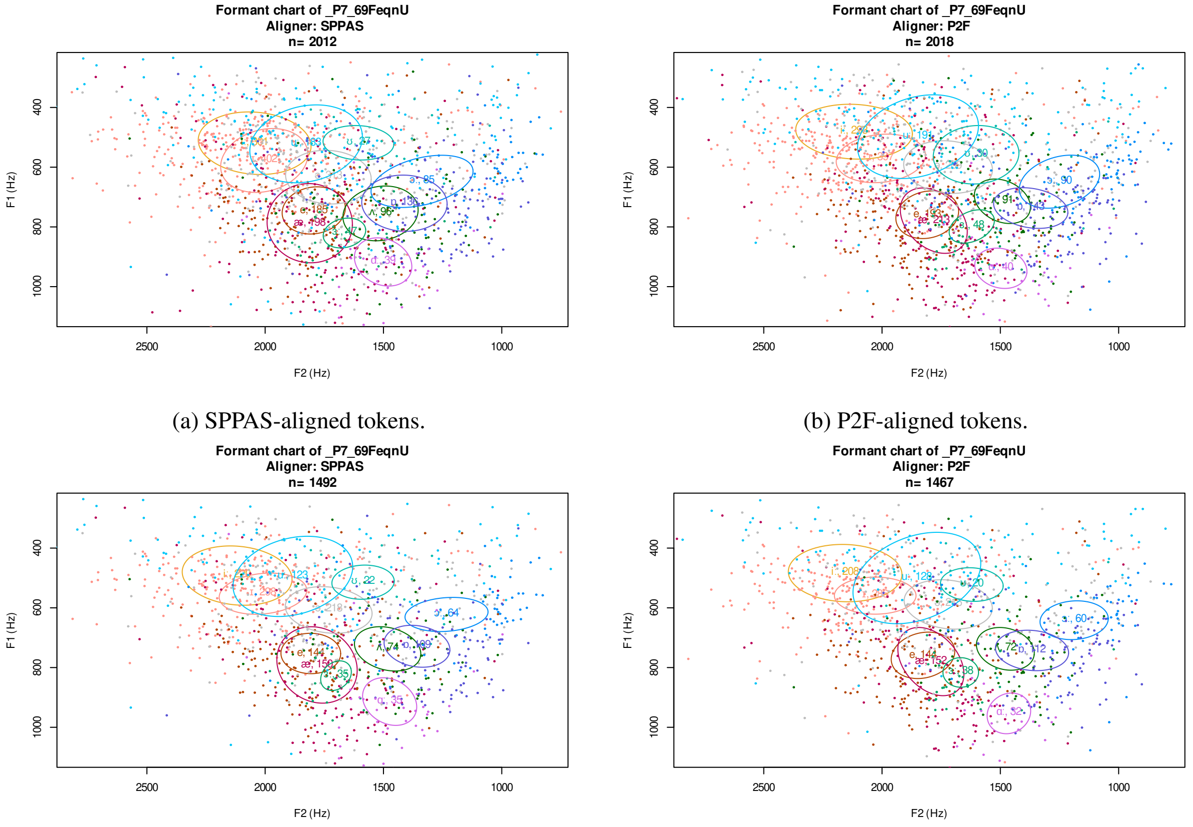
The YouTube Phonetics Pipeline is a Python-based series of scripts for the automatic extraction of audio (or video) content from YouTube and other streaming services. Its main characteristics are described in Coats (2023c). Like PEASYV, YTPP makes use of the open-source yt-dlp library for harvesting YouTube's automatic speech recognition transcripts and audiovisual content; transcripts are then aligned using the Montreal Forced Aligner (MFA) (McAuliffe et al., 2017a). The output from the aligner, in the TextGrid format, is then sent to Parselmouth-Praat (Jadoul et al., 2018; Boersma and Weenink, 2023), a Python port of functions from Praat. This approach allows for the automated analysis of vowel formants, pitch, prosody, or other acoustic parameters within the functionality of a Jupyter notebook. The basic methods of YTPP are available in a Colab environment.[4] Because YTPP is developed in a Jupyter environment, it is fully modifiable, and data can be analyzed statistically or visualized for exploratory analysis with widely used libraries, according to user needs. Transcript data for several publicly available corpora has been collected using the basic approach employed by YTPP (Coats, 2023a).

YTTP was used to extract F1 and F2 formant values for monophthongs from the YouTube test video noted above in Section 3. Figure 4 depicts the vowel space for the video _P7_69FeqnU, entitled "Sentence Stress and Intonation in English" from the *Pronunciation with Emma* channel, using an acoustic models trained on UK English, a pronunciation dictionary for UK English, and a phoneset meant to represent UK English.[5] As in Figure 3,

[3] Current information is at https://www.adrienmeli.xyz/peasyv.html

[4] https://github.com/stcoats/phonetics_pipeline

[5] English (UK) MFA dictionary v2.2.1; English MFA acoustic model v2.2.1, https://mfa-models.



(a) SPPAS-aligned tokens.

(b) P2F-aligned tokens.

(c) Matching SPPAS-aligned tokens.

(d) Matching P2FA-aligned tokens.

Figure 3: PEASYV flowcharts of video _P7_69FeqnU.

the centers of the circles represent the mean measurement values for the monophthong vowels in F1/F2 formant space and the ellipses values within one standard deviation of the mean values; the IPA symbol for each vowel is followed by the number of vowel tokens detected by the aligner in the video.[6]

Figure 4 differs somewhat from Figure 3, not only due to different plotting software being employed, but also due to differences between the acoustic models and phonemic representations in the three systems under consideration. Nevertheless, the figures suggest that the speaker in the video, as the name of her channel suggests, has vowels that correspond to standard English pronunciation norms. Future work may undertake more careful comparison of these (and other aligners) by controlling for the acoustic models employed by the different algorithms and the underlying graphemic representations.

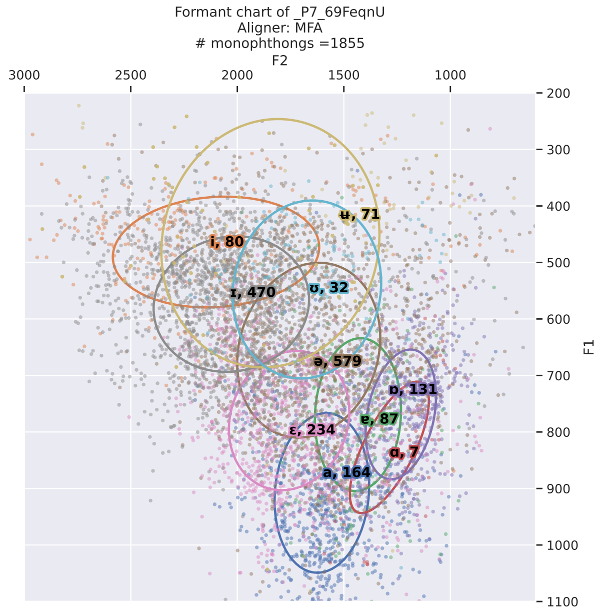


Figure 4: YTPP formant chart for the video _P7_69FeqnU

readthedocs.io/en/latest/acoustic/index.html. MFA's functionality includes a variety of acoustic models, dictionaries, phonesets, and other options.

[6] In this example, the script has set the number of measurements per phone at 9, at equally spaced intervals within the total duration of the phone, but formant intensity could not be registered at all measurement intervals due to acoustic quality. The number of measurements per phone can be changed in the script.

Python plotting functionality can also be used to generate Praat-style charts of sound intensity and frequency, as in Figure 5.

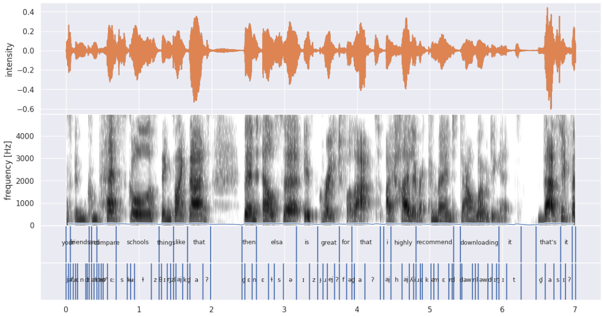


Figure 5: Sound intensity and frequency for an excerpt of _P7_69FeqnU

5 Discussion, caveats, and outlook

No longer must the phonetician travel to distant locales with a tape recorder and painstakingly interview informants: both PEASYV and YTTP offer researchers in phonetics and acoustic analysis the means for the automatic and extraction and analysis of hundreds or thousands of hours of speech.

PEASYV output grids include the results of two aligners: the overlap method described above may help to identify and extract segments more accurately, especially for audio files with acoustic background noise. PEASYV also includes syllabification information, making it potentially useful for automated studies of lexical stress patterns or other prosodic features.

YTPP utilizes the MFA aligner, which is more recent and possibly more accurate than HTK- or Julius-based aligners (see the citations above). In addition, YTPP is available and can already be used "out-of-the-box" for data collection and analysis tasks. Its code is fully available and customizable.

The pipelines both offer the means to collect and analyze online speech recordings, but two considerations should be noted pertaining to the accuracy of ASR transcripts and the legal contexts in which online data collection can be undertaken.

5.1 ASR Accuracy

While ASR has made great advances in recent years, many ASR transcripts of videos on YouTube (and other platforms) contain errors due to issues such as poor audio quality, out-of-vocabulary lexical items, or strongly accented speech not accounted for in the training data. Despite this, given sufficient quantities of data, transcript errors in phonetic analysis pipelines such as PEASYV and YTPP may tend to cancel each other out: Coto-Solano (2022), for example, found that even pipelines that utilize error-ridden transcripts are generally able to accurately capture the formant values of a given speaker.

5.2 Legal context

While content from YouTube and other streaming platforms is generally owned by the content creator and/or the platform, use of copyrighted content for non-profit purposes such as academic research is generally permitted in most jurisdictions. In the US, for example, the "Fair Use" provisions of copyright law (U.S.C. Title 17, § 107) permit reuse of copyrighted material for research purposes; other Anglophone jurisdictions have similar laws.

In the EU, Directive 2019/790 of the European Parliament and of the Council instructed member states to pass legislation allowing the re-use of copyrighted content for purposes of scientific research or teaching; the directive has since been implemented by most member state legislatures (see also the discussion in Coats).

We expect that legislation will continue to permit fair and reasonable use of copyrighted materials for non-profit research purposes and that researchers who follow the appropriate ethical guidelines will be able to make use of PEASYV and YTPP for data collection.

5.3 Outlook

A paradigm shift in data collection and analysis practices in the language sciences is underway, and PEASYV and YTPP represent potentially valuable tools for researchers in a wide variety of linguistic subfields. Future work with the pipelines may include, as noted above, more detailed comparison of aligners and of outputs; the development of interoperability with other data formats (for example, PolyglotDB McAuliffe et al. 2017b, an SQL-based system with a Python API for the organization of speech data and alignments); and the creation of searchable online databases that include aligned audio content. In a broader perspective, it is hoped that the tools will help researchers to collect and study the rich acoustic variation of the speech signal.

References

Brigitte Bigi. 2012. SPPAS: a tool for the phonetic segmentations of speech. In *The Eighth International Conference on Language Resources and Evaluation*, pages 1748–1755.

Paul Boersma and David Weenink. 2023. Praat: doing phonetics by computer [Computer program]. Version 6.3.10, retrieved 1 June 2023 `http://www.praat.org/`.

Steven Coats. 2023a. Dialect corpora from Youtube. In *Language and linguistics in a complex world*, pages 79–102. De Gruyter.

Steven Coats. 2023b. A new corpus of geolocated ASR transcripts from Germany. *Language Resources and Evaluation*.

Steven Coats. 2023c. A pipeline for the large-scale acoustic analysis of streamed content. In *Proceedings of the 10th International Conference on CMC and Social Media Corpora for the Humanities (CMC-Corpora 2023)*, page 51–54. Mannheim: Leibniz-Institut für Deutsche Sprache.

Rolando Coto-Solano. 2022. Evaluating word embeddings in extremely under-resourced languages: A case study in Bribri. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4455–4467, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Simon Gonzalez, James Grama, and Catherine E Travis. 2020. Comparing the performance of forced aligners used in sociophonetic research. *Linguistics Vanguard*, 6(1):20190058.

Daniel Hirst and Robert Espesser. 1993. Automatic modelling of fundamental frequency using a quadratic spline function. *Travaux de l'Institut de Phonétique d'Aix*, pages 75–85.

Daniel J Hirst. 2007. A Praat plugin for Momel and INTSINT with improved algorithms for modelling and coding intonation. In *Proceedings of the XVIth International Conference of Phonetic Sciences*, pages 1233–1236.

Yannick Jadoul, Bill Thompson, and Bart de Boer. 2018. Introducing Parselmouth: A Python interface to Praat. *Journal of Phonetics*, 71:1–15.

Thomas Kisler, Uwe Reichel, and Florian Schiel. 2017. Multilingual processing of speech via web services. *Computer Speech & Language*, 45:326–347.

Akinobu Lee and Tatsuya Kawahara. 2019. julius-speech/julius: Release 4.5.

Laurel MacKenzie and Danielle Turton. 2020. Assessing the accuracy of existing forced alignment software on varieties of British English. *Linguistics Vanguard*, 6(s1):20180061.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017a. Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In *Proc. Interspeech 2017*, pages 498–502.

Michael McAuliffe, Elias Stengel-Eskin, Michaela Socolof, and Morgan Sonderegger. 2017b. Polyglot and speech corpus tools: A system for representing, integrating, and querying speech corpora. In *INTERSPEECH*, pages 3887–3891.

Adrien Méli and Nicolas Ballier. 2023. PEASYV: A procedure to obtain phonetic data from subtitled videos. *Proceedings of the International Congress of Phonetic Sciences*, pages 3211 – 3215.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The Kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, CONF. IEEE Signal Processing Society.

Robert L. Weide. 1994. The CMU Pronouncing Dictionary. *http://www.speech.cs.cmu.edu/cgi-bin/cmudict*.

John C. Wells. 2008. *Longman Pronunciation Dictionary*. Pearson Longman, London.

Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, et al. 2006. *The HTK Book, version 3.4*. Cambridge University Engineering Department, Cambridge, UK.

Direct Speech to Text Translation: Bridging the Modality Gap Using SimSiam

Balaram Sarkar
IIT Indore
Madhya Pradesh, India
ms2204101006@iiti.ac.in

Chandresh Kumar Maurya
IIT Indore
Madhya Pradesh, India
chandresh@iiti.ac.in

Anshuman Agrahri
IIT Indore
Madhya Pradesh, India
phd2201101008@iiti.ac.in

Abstract

Learning similar representations for spoken utterances and their written text involves understanding both forms in a shared manner. This process of developing similar representations for semantically related speech and text is essential, particularly for tasks like speech-to-text (S2T) translation. To that end, we propose a SimSiam-based S2T (**S3T**) model that leverages the SimSiam network, a state-of-the-art unsupervised learning architecture, to bridge the modality gap between speech and text. The proposed model does not require negative sample mining. The comparative study using four directions of the standard MuST-C (Di Gangi et al., 2019) dataset demonstrates that the proposed S3T translation model beats all the existing methods, and achieves an average metric of 30.02 BLEU score. Our analysis affirms that S3T effectively bridges the representation gap between the two modalities.

1 Introduction

Speech-to-text (S2T) translation is to map speech input in a given language to text output in another language. It has applications in video subtitling, facilitating communication across different demographics, education, etc. Traditional approaches for solving S2T tasks cascade two models: machine translation (MT) and automatic speech recognition (ASR). Cascade models suffer from high latency, error propagation, and memory cost. Therefore, recent works addressing S2T use end-to-end (E2E) models based on pre-trained models such as (Inaguma et al., 2020; Bérard et al., 2018; Wang et al., 2020b; Bansal et al., 2019; Le et al., 2021) or multi-task learning (joint-training) approaches (Chuang et al., 2020; Anastasopoulos and Chiang, 2018; Wang et al., 2019; Ye et al., 2022; Sperber et al., 2019; Le et al., 2020; Tang et al., 2021b). A very recent work (Ye et al., 2022) hypothesizes that the low performance of E2E models is due to the modality gap between speech and text representations. Building on the same hypothesis, we present a novel methodology based on the SimSiam (Chen and He, 2021) network, leveraging the cosine similarity (CS) loss, to mitigate the modality gap between speech and textual representations. Unlike (Ye et al., 2022), the proposed model learns joint representations in an unsupervised way and does not need negative sample mining. Our major contributions are given as follows: (a) We utilize SimSiam architecture to reduce the modality gap between textual and speech representation for the first time. As per our knowledge, such a study has not been done before, and (b) Empirical results on benchmark MuST-C data show the superiority of our approach where it outperforms the baseline by 0.17 BLEU score. Analysis indicates that the proposed approach is able to fill the modality gap.

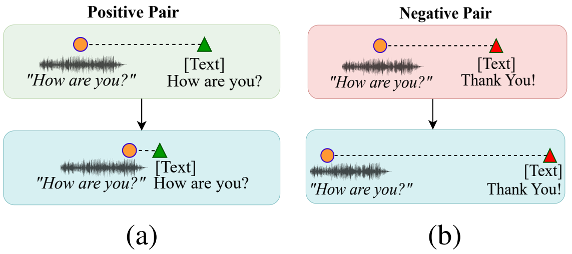


Figure 1: Depiction of representations for speech and textual transcripts. An ideal representation is where two different modalities with the same meaning (positive pair) should be close to each other as shown in (a) and it's the opposite for negative pairs in (b).

2 Related Work

Our work jointly studies end-to-end (E2E) S2T tasks and methods to counter the modality gap between speech and text.

2.1 Speech-to-Text

The traditional approach to solve S2T problem consists of cascaded systems using an ASR followed

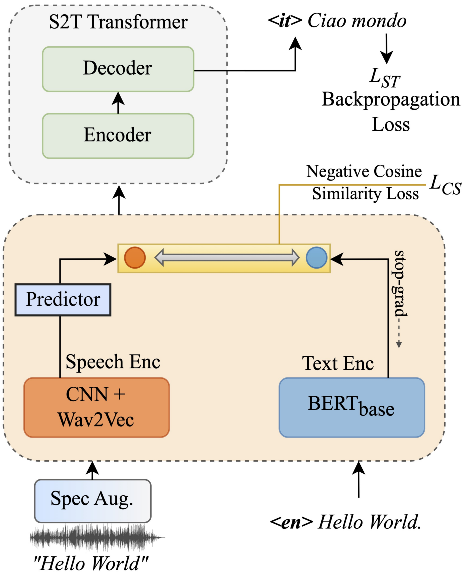


Figure 2: Proposed model architecture.

by an MT module. This method still has some limitations such as being susceptible to error propagation and having high latency (Anastasopoulos and Chiang, 2018). Recently, various authors have explored the end-to-end S2T models (Le et al., 2020; Weiss et al., 2017; Tang et al., 2022a; Di Gangi et al., 2019; Inaguma et al., 2020). Earlier, major work in this domain only produced modest results for S2T data (Tang et al., 2022a; Weiss et al., 2017), whereas the current work approached the results of the cascaded S2T models closely (Ye et al., 2022, 2021; Bentivogli et al., 2021; Xu et al., 2021; Tang et al., 2021a).

2.2 Speech and Text Alignment

The previous S2T models have worked on aligning text and speech embeddings, e.g., using an adversarial loss (Alinejad and Sarkar, 2020) in supervised pre-training, in self-supervised pre-training (Ao et al., 2022; Chen et al., 2022; Bapna et al., 2021), and using Euclidean distance (Dong et al., 2021; Liu et al., 2020; Tang et al., 2021a), cosine distance (Chuang et al., 2020), Kullback–Leibler divergence (Tang et al., 2022b), and contrastive loss (Han et al., 2021; Ye et al., 2022; Ouyang et al., 2022) in multi-task learning. All these methods require negative samples from the corpus to train the model, whereas our approach works without the need for any negative sample.

3 Problem Definition

The problem of S2T is defined as follows. Given the sequence of input audio features $x = (x_1, \ldots, x_{|x|})$ and its transcript $t = (t_1, \ldots, t_{|t|})$, the goal is to learn a representation as shown in Figure 1. More formally, the cosine similarity (CS) between positive pairs of speech and text representations (x_p, t_p) is less than the CS between negative pairs (x_n, t_n) in the embedding space.

$$\mathrm{CS}(f(x_p), f(t_p)) < \mathrm{CS}(f(x_n), f(t_n)) \quad (1)$$

Where f is the representation learning function. The new representations are used for the downstream S2T task, where the S2T model seeks to optimize the following objective function:

$$\theta^* = \arg\max_{\theta} \mathcal{L}(f(x, t), y) \quad (2)$$

Where $\mathcal{L}(\cdot)$ denotes the loss function of the S2T model and $y = (y_1, \ldots, y_{|y|})$ is the sequence of target text translations.

4 Method

The S2T baseline used to optimize the objective function (2) is a transformer-based encoder-decoder model. The core idea behind our approach is to use CS to align the source speech and transcript pairs and use it for downstream S2T tasks. The hypothesis is that source speech and corresponding transcript representations should be closer in the embedding space since they represent the same semantics. To that end, we seek to employ the approach originally proposed for visual recognition task handling similarity learning using SimSiam (Chen and He, 2021). Motivated by its recent application, we ask the following research question: Will the same approach be able to learn similar representations in an S2T setting? We confirm that using Siamese-like encoders for speech and transcript in an earlier stage can yield better results for the S2T task and help bridge the modality gap without negative sample mining.

4.1 SimSiam Network

Our main goal is to reduce the modality gap in S2T, which arises due to the distance between speech and textual representations. To propose a solution for this issue, we introduce an architecture influenced by (Chen and He, 2021) comprising two encoders as shown in Figure 2: One for speech and the other for text input.

$$H \triangleq (h_1, \ldots, h_{|x|}) \triangleq \mathtt{ENCODE}(x; \theta_m)$$
$$K \triangleq (k_1, \ldots, k_{|t|}) \triangleq \mathtt{ENCODE}(t; \theta_n)$$

where H and K are the hidden feature vectors of audio speech sequences and their transcripts, and θ_m and θ_n are the parameters of the text and speech encoders respectively. We use Wav2Vec (Baevski et al., 2020) followed by CNN as speech encoder and as the text encoder we use a BERT base uncased (Devlin et al., 2019) model. The input pair of speech x and its parallel text t are fed to the corresponding encoders as shown in Figure 2. The SimSiam network is trained by minimizing negative cosine similarity in an unsupervised manner to generate features that are close to each other in the embedding space. The gradients from the text encoder's contribution to the loss are not used to update the speech encoder's parameters in (3) and vice versa, and this is achieved by applying the stop-gradient ($\mathtt{SG}$) operation. We utilize $\mathtt{SG}$ with symmetric CS loss defined as follows:

$$\mathcal{L}_{CS} = \frac{1}{2}\mathcal{D}(H, \mathtt{SG}(K)) + \frac{1}{2}\mathcal{D}(K, \mathtt{SG}(H)) \quad (3)$$

This allows the model to learn more meaningful features from the input data.

4.2 S2T Transformer

The S2T Transformer model is a variant of the Transformer architecture adapted for processing the aligned speech-text representation as input. These features are passed through the S2T encoder containing multiple layers of self-attention mechanisms that allow the model to process different parts of the input sequence and effectively capture long-range dependencies. A self-attention mechanism computes attention weights to emphasize important features while decoding the output. During training, the model is typically tuned to a ground truth target transcript of the spoken audio by optimizing the following loss function:

$$\mathcal{L} = \mathcal{L}_{CS} + \mathcal{L}_{ST} \quad (4)$$

where

$$\mathcal{L}_{ST} = -\sum_{n} \log P(x_n | y_n)$$

$\mathcal{L}_{ST}$ is the label-smoothed-cross-entropy loss on <*speech, target text*> pairs. The output of the S2T transformer is a sequence of predicted tokens representing the translated text.

Methods	BLEU				
	De	Fr	Nl	It	Avg
NeurST	22.8	33.3	27.2	22.9	26.55
ESPnet-ST	22.9	32.7	27.4	23.8	26.7
Dual-decoder	23.6	33.5	27.6	24.2	27.22
FAIRSEQ S2T	24.5	34.9	28.6	24.6	28.15
XSTNet	25.5	36	30	25.5	29.25
ConST	25.7	36.8	**30.6**	**26.3**	29.85
S3T	**26.8**	**37**	30.2	26.1	**30.02**

Table 1: Performance of baselines and proposed model on MuST-C test split.

5 Experiment

In this section, we explain the (a) datasets, (b) baselines, (c) training and testbed followed by (d) metrics used during the evaluation.

5.1 Dataset

We conduct experiments on four pairs of translation directions available in **MuST-C**[1] (Di Gangi et al., 2019) dataset: English (En) to German (De), French (Fr), Dutch (Nl) and Italian (It). It contains audio, transcript and translation from TED talks for each direction.

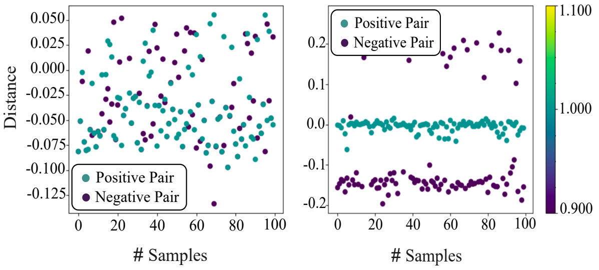
5.2 Baselines

We compare our model with two kinds of baseline: (1) standard E2E S2T models, and (2) E2E S2T models with modality bridging techniques. In the first category, we compare performance with NeurST (Zhao et al., 2021), ESPNet-ST, S2T with Dual Decoder, FAIRSEQ-S2T, and XSTNet (Ye et al., 2021). For the second category, we compare with ConST which uses contrastive loss to attract positive pairs and repel negative pairs. Note that such a scheme requires negative sample mining which is costly.

5.3 Training and Testbed

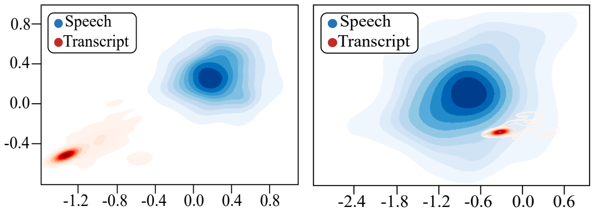
The method in this work is implemented using FAIRSEQ S2T toolkit (Wang et al., 2020a). The backbone framework consists of an S2T Transformer encoder-decoder model as shown in Figure 2. The number of self-attention layers for both the encoder and decoder is set to 6, with 8 attention

[1] We use v1.0. `https://ict.fbk.eu/must-c/`



(a) (b)

Figure 3: Scatter plot showing distances between positive and negative speech-text pairs (a) before, and (b) after training. The positive and negative pairs form separate clusters.



(a) (b)

Figure 4: Bivariate KDE contour plot for the embeddings of English speech and text(a) before and (b) after training. The red lines denote the text and the blue lines denote the speech representations.

heads in each layer. Due to training resource constraints, the encoder and decoder architecture is $medium$ and consists of 512 hidden units. The training is halted when the performance is not improved for 15 consecutive epochs. The SpecAugment (Park et al., 2019) is used for data augmentation, and the GELU activation function is used to shift normalization and improve convergence and training stability. The S2T model is trained using label-smoothed-cross-entropy loss with a value of 0.1 as the label smoothing factor. Adam optimizer with a learning rate of 1e-4, and the learning rate schedule using an inverse square root scheduler was used.

5.4 Performance Metric

Case-sensitive detokenized BLEU using sacreBLEU is used to report the performance of the model. We average the ten best checkpoints and predict the output using a beam size of five. All experiments are repeated with three different random seeds, and we report the average BLEU on the MuST-C `tst-COMMON` set.

6 Results

This section presents the results of the comparative evaluation followed by an analysis of our proposed method.

6.1 Comparative Evaluation

Table 1 shows the main results. We compare our method with several S2T baselines. Many existing works utilize external data, such as ASR/MT data, to boost their model performance. We include models without external MT data for fair comparison and compare results with the model's *medium* architecture due to computational constraints. Comparison with standard E2E S2T models shows that our method consistently outperforms in all directions with an average BLEU of 30.02. Compared to ConST, the proposed method outperforms in two directions (De and Fr) and achieves a gain in average BLEU score of 0.17. Additionally, our approach does not need to mine any negative samples as ConST does.

6.2 Analysis

The effectiveness of our approach is shown in Figure 3. Although our method works without any positive or negative sample mining, we aim to determine its capacity to distinguish between positive and negative pairs without requiring explicit labeling. We plot the distance between pairs of speech and text samples (positive pairs with the same meaning and negative ones with different meanings) before and after the model is trained. It shows a reduction in the distance between the positive pair of samples and an increase in the distance between the negative pair of samples. To look more into it, the bivariate kernel density estimation (Parzen, 1962) (KDE) contour of the features are plotted as shown in Figure 4. If the speech and its parallel text embeddings are similar, their contour lines will overlap as much as possible. As shown in Figure 4(b), the proposed method is able to align the two representations and close the gap.

7 Conclusion

We propose S3T, a S2T framework bridging the speech-text modality gap in an unsupervised way. Results on MuST-C indicate the effectiveness of the proposed method compared to baselines. Future works may explore designing even better modality bridging techniques leveraging external data.

Limitations

Although our proposed method outperforms most baselines on the S2T benchmark, it still has some limitations: (1) the choice of hyperparameters such as learning rates, batch sizes, and the length of the projection network can significantly impact the training process and the quality of learned representations, so we need to make careful choices about it's settings; (2) with a smaller dataset, this approach might not work as effectively, because there is less variety and fewer examples for the model to learn from during training; (3) how to apply our method to other tasks also needs to be studied further.

References

Ashkan Alinejad and Anoop Sarkar. 2020. Effectively pretraining a speech translation decoder with machine translation data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8014–8020.

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. 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 82–91.

Junyi Ao, Rui Wang, Long Zhou, Chengyi Wang, Shuo Ren, Yu Wu, Shujie Liu, Tom Ko, Qing Li, Yu Zhang, Zhihua Wei, Yao Qian, Jinyu Li, and Furu Wei. 2022. SpeechT5: Unified-modal encoder-decoder pre-training for spoken language processing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5723–5738.

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *CoRR*, abs/2006.11477.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2019. Pre-training on high-resource speech recognition improves low-resource speech-to-text 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 58–68.

Ankur Bapna, Yu-An Chung, Nan Wu, Anmol Gulati, Ye Jia, Jonathan H. Clark, Melvin Johnson, Jason Riesa, Alexis Conneau, and Yu Zhang. 2021. SLAM: A unified encoder for speech and language modeling via speech-text joint pre-training. *CoRR*, abs/2110.10329.

Luisa Bentivogli, Mauro Cettolo, Marco Gaido, Alina Karakanta, Alberto Martinelli, Matteo Negri, and Marco Turchi. 2021. Cascade versus direct speech translation: Do the differences still make a difference? 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 2873–2887.

Alexandre Bérard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin. 2018. End-to-end automatic speech translation of audiobooks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6224–6228.

Xinlei Chen and Kaiming He. 2021. Exploring simple siamese representation learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15745–15753.

Zhehuai Chen, Yu Zhang, Andrew Rosenberg, Bhuvana Ramabhadran, Pedro J. Moreno, Ankur Bapna, and Heiga Zen. 2022. Maestro: Matched speech text representations through modality matching. In *Interspeech*.

Shun-Po Chuang, Tzu-Wei Sung, Alexander H. Liu, and Hung-yi Lee. 2020. Worse WER, but better BLEU? leveraging word embedding as intermediate in multitask end-to-end speech translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5998–6003.

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.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. MuST-C: a Multilingual Speech Translation Corpus. 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 2012–2017.

Qianqian Dong, Rong Ye, Mingxuan Wang, Hao Zhou, Shuang Xu, Bo Xu, and Lei Li. 2021. Listen, understand and translate: Triple supervision decouples end-to-end speech-to-text translation. In *AAAI Conference on Artificial Intelligence*.

Chi Han, Mingxuan Wang, Heng Ji, and Lei Li. 2021. Learning shared semantic space for speech-to-text translation. *CoRR*, abs/2105.03095.

Hirofumi Inaguma, Shun Kiyono, Kevin Duh, Shigeki Karita, Nelson Yalta, Tomoki Hayashi, and Shinji Watanabe. 2020. ESPnet-ST: All-in-one speech translation toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 302–311.

Hang Le, Juan Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2021. Lightweight Adapter Tuning for Multilingual Speech Translation. In *The Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2021)*.

Hang Le, Juan Miguel Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2020. Dual-decoder transformer for joint automatic speech recognition and multilingual speech translation. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, pages 3520–3533.

Yuchen Liu, Junnan Zhu, Jiajun Zhang, and Chengqing Zong. 2020. Bridging the modality gap for speech-to-text translation. *CoRR*, abs/2010.14920.

Siqi Ouyang, Rong Ye, and Lei Li. 2022. WACO: Word-Aligned Contrastive Learning for Speech Translation. *arXiv e-prints*, page arXiv:2212.09359.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin Dogus Cubuk, and Quoc V. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. In *Interspeech*.

Emanuel Parzen. 1962. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065 – 1076.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2019. Attention-passing models for robust and data-efficient end-to-end speech translation. *Transactions of the Association for Computational Linguistics*, 7:313–325.

Yun Tang, Hongyu Gong, Ning Dong, Changhan Wang, Wei-Ning Hsu, Jiatao Gu, Alexei Baevski, Xian Li, Abdelrahman Mohamed, Michael Auli, and Juan Pino. 2022a. Unified speech-text pre-training for speech translation and recognition. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1488–1499.

Yun Tang, Hongyu Gong, Ning Dong, Changhan Wang, Wei-Ning Hsu, Jiatao Gu, Alexei Baevski, Xian Li, Abdelrahman Mohamed, Michael Auli, and Juan Pino. 2022b. Unified speech-text pre-training for speech translation and recognition. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1488–1499.

Yun Tang, Juan Pino, Xian Li, Changhan Wang, and Dmitriy Genzel. 2021a. Improving speech translation by understanding and learning from the auxiliary text translation 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 4252–4261.

Yun Tang, Juan Pino, Changhan Wang, Xutai Ma, and Dmitriy Genzel. 2021b. A general multi-task learning framework to leverage text data for speech to text tasks. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6209–6213.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020a. Fairseq S2T: Fast speech-to-text modeling with fairseq. 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: System Demonstrations*, pages 33–39.

Chengyi Wang, Yu Wu, Shujie Liu, Zhenglu Yang, and M. Zhou. 2019. Bridging the gap between pre-training and fine-tuning for end-to-end speech translation. In *AAAI Conference on Artificial Intelligence*.

Chengyi Wang, Yu Wu, Shujie Liu, Ming Zhou, and Zhenglu Yang. 2020b. Curriculum pre-training for end-to-end speech translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3728–3738.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Z. Chen. 2017. Sequence-to-sequence models can directly translate foreign speech. In *Interspeech*.

Chen Xu, Bojie Hu, Yanyang Li, Yuhao Zhang, Shen Huang, Qi Ju, Tong Xiao, and Jingbo Zhu. 2021. Stacked acoustic-and-textual encoding: Integrating the pre-trained models into speech translation encoders. 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 2619–2630.

Rong Ye, Mingxuan Wang, and Lei Li. 2021. End-to-end speech translation via cross-modal progressive training. pages 2267–2271.

Rong Ye, Mingxuan Wang, and Lei Li. 2022. Cross-modal contrastive learning for speech translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5099–5113.

Chengqi Zhao, Mingxuan Wang, Qianqian Dong, Rong Ye, and Lei Li. 2021. NeurST: Neural speech translation toolkit. 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 55–62.

Improving Dhivehi Automatic Speech Recognition (ASR) with Sub-word Modelling, Language Model Decoding and Automatic Spelling Correction

Arushad Ahmed
University of St. Andrews, St Andrews KY16 9AJ, United Kingdom
aa369@st-andrews.ac.uk

Abstract

Current state-of-the-art (SOTA) Automatic Speech Recognition (ASR) models are multilingual. While these models have greatly improved transcription accuracy for high-resourced languages, under-resourced languages still require further language specific optimizations and finetuning to achieve acceptable levels of accuracy. In this work we explore ways of improving ASR for Dhivehi, an under-resourced South Asian language, by finetuning pretrained multilingual ASR models, Sub-word Modelling, Language Model (LM) decoding and Automatic Spelling Correction. We finetune 5 Dhivehi ASR models and apply our accuracy boosting techniques, with one of our models achieving a new state-of-the-art Word Error Rate (WER) of 14.26% on the Dhivehi Common Voice ASR benchmark, which is a 31.93% relative WER improvement over the existing SOTA of 20.95%. We create a new Dhivehi text corpus, and train 2 new Dhivehi LMs to support our accuracy boosting techniques.

1 Introduction

Recent work on Automatic Speech Recognition (ASR) has focused on training multilingual models; (Zhang, et al., 2023; Hou, et al., 2020a; Pratap, et al., 2023; Conneau, et al., 2020; Radford, et al., 2022). While these multilingual models have produced state-of-the-art (SOTA) results for high-resource languages, results (Hou, et al., 2020a) show that, especially for low-resource languages, there is room for improving transcription accuracy using language specific optimizations and fine-tuning. In this work, we focus on improving ASR accuracy for Dhivehi (ދިވެހި), the native language of the Maldives.

Modern ASR models follow all-neural architectures that enable End-to-End (E2E) speech recognition by training directly on audio recordings and producing text transcriptions as output (Prabhavalkar, et al., 2023). E2E ASR models either explicitly or implicitly align the output text to the input audio.

Connectionist Temporal Classification (CTC) (Graves, et al., 2006) based E2E models use explicit alignment by assigning an output text token to each element of the input audio sequence (Hannun, 2017). CTC models simply learn a mapping from aspects of speech such as phonemes and diphones to output character sequences (Prabhavalkar, et al., 2023; Hannun, 2017). Therefore, CTC models can benefit from sub-word modelling of the output vocabulary, which better corresponds with the phonemes of the target language (Xu, et al., 2019; Zhou, et al., 2021). Moreover, CTC model accuracy can be further improved by incorporating the probabilities from a Language Model (LM) into the output decoding process, which can implicitly model the syntax and semantics of the language (Baevski, et al., 2020). The outputs generated by LM decoding can be further improved by Automatic Spelling Correction (Zhang, et al., 2019).

Attention-Based Encoder-Decoder (AED) ASR models contain an Encoder which produces context vectors using the input acoustic frames and a Decoder which uses the Attention Mechanism (Bahdanau, et al., 2016) to generate a text sequence from the context vectors, without explicitly aligning the text and audio (Prabhavalkar, et al., 2023). AED models implicitly learn a language model over the training outputs (Prabhavalkar, et al., 2023). As a result, they benefit less from external LMs and spelling correction.

E2E models pretrained on high-resourced languages can be fine-tuned on under-resourced languages like Dhivehi through transfer learning to give better results than training from scratch

(Baevski, et al., 2020; Conneau, et al., 2020; Radford, et al., 2022; Pratap, et al., 2023; Hou, et al., 2020b). In this work, we finetune pretrained XLSR (Conneau, et al., 2020), MMS (Pratap, et al., 2023) and Whisper (Radford, et al., 2022) models on Dhivehi speech data from the Mozilla Common Voice 13.0 (CV-13) (Ardila, et al., 2019) dataset and experiment with applying different accuracy boosting techniques.

2 Related Work

Tyers & Meyer (2021) used the Coqui STT [1] toolkit to train ASR models for several different under-resourced languages including Dhivehi. They used transfer learning by finetuning an English ASR model based on the Mozilla Deep Speech architecture, which is an open-source implementation of the Deep Speech ASR architecture from Baidu (Hannun, et al., 2014). They used hyperparameter search to optimize the model hyperparameters for specific languages (Tyers & Meyer, 2021). LM decoding was utilized to further boost the results. For Dhivehi, the authors used 3:56:12 of training data from Common Voice (CV) [2] for the ASR model and 6.8MB of text containing 419k tokens of 76k different types for LM training. For Dhivehi, the authors obtained a WER of 88.37% without LM and 66.49% with LM decoding (Tyers & Meyer, 2021).

Pham, et al. (2021) trained multi-lingual ASR models for 27 languages (including Dhivehi) based on the Transformer architecture (Vaswani, et al., 2023) and LSTM architectures (Hochreiter & Schmidhuber, 1997) using a novel weight factorization scheme for efficient multi-lingual training. The models contained weights shared across all the languages as well as language specific adapter layers (Pham, et al., 2021). Under this work, the best result obtained for Dhivehi was a WER of 63.72% using the weight factorized version of the Transformer based model (Pham, et al., 2021).

Hou, et al. (2020a), trained multilingual ASR models for 42 languages based on a hybrid CTC/attention architecture using 5,000 hours of speech. One of these models were trained using a character level output vocabulary while the other was trained using a sub-word vocabulary (Hou, et al., 2020a). It was shown that, generally the larger sub-word vocabulary produced better results across all 42 languages. After training these multilingual models, transfer learning was used to finetune ASR models for 14 different low-resource languages including Dhivehi. For each of these 14 languages, a language specific model was finetuned as well as a joint 14-language multilingual model. The authors experimented with finetuning these models using the pretrained models as well as training the models from scratch. For Dhivehi, the authors had used 6 hours of training data from CV. The best result obtained was a WER of 54.7% using the Dhivehi specific model with pretraining (Hou, et al., 2020a).

Hassaan, et al. (2018) trained Dhivehi ASR models using CMUSphinx speech recognition toolkit. These models were Hidden Mark Model (HMM) based acoustic models which were boosted using N-gram LMs (Hassaan, et al., 2018). The authors had initially trained a model to recognize spoken numerals in Dhivehi which had a reported accuracy of 75%. They also trained another model to recognize general Dhivehi speech, which had a reported accuracy of 42.5%. The LM used by the authors was trained on 600MB of Dhivehi text scraped from the web and the acoustic model was trained on 48:33 of speech data collected through a web interface, mobile app, and Telegram bot that the authors created, which presented users with samples from the text corpus that were read and recorded.

Apart from formal work conducted on Dhivehi ASR, there has been some personal work done on the topic by some individuals as well. The most notable of these works are the Dhivehi ASR models trained by Shahuza Abdul Kareem [3] and published on Hugging Face. Specifically, her Wav2Vec2-XLS-R-1B-dv model [4] has the best reported WER of any Dhivehi ASR model known publicly so far. This model is a finetuned version of Facebook's Wav2Vec2-XLS-R-1B checkpoint (Conneau, et al., 2020) which follows the hybrid CNN/Transformer/CTC architecture introduced in Baevski, et al. (2020). The model was trained using

[1] https://github.com/coqui-ai/STT

[2] https://commonvoice.mozilla.org/

[3] https://huggingface.co/shahukareem?sort_models=downloads#models

[4] https://huggingface.co/shahukareem/wav2vec2-xls-r-1b-dv

around 25 hours of speech from Common Voice (Ardila, et al., 2019) version 8.0 with a character-based output vocabulary. The reported WER was 21.23% on Common Voice 8.0 evaluation set, which can be considered as the state-of-the-art for Dhivehi ASR.

3 Design

3.1 Pretrained ASR Models

Based on the results from previous works, a transfer learning approach of finetuning pretrained models was chosen instead of training models from scratch. The models chosen for finetuning were pretrained multilingual models from recent works that had claimed state-of-the-art performance results on common English ASR benchmarks. Here we will list and discuss the architectures of these models.

XLSR: XLSR (Conneau, et al., 2020) is a model pretrained on 53 languages following the Hybrid CTC architecture of (Baevski, et al., 2020). This architecture consists of a Convolution Neural Network (CNN) based feature extractor which extracts log-mel features from the input audio which are quantized through product quantization (Baevski, et al., 2020). The feature vectors are then passed into a Transformer based encoder network, which learns context vectors from these feature vectors using Contrastive Loss (Baevski, et al., 2020). For fine tuning of the model, the CNN layers can be frozen and a linear output layer corresponding to the desired output vocabulary can be initialized on top of the encoder network and trained using CTC loss (Baevski, et al., 2020). For the XLSR model, the authors were able to demonstrate that pretraining the encoder on a large number of languages produced improved performance when finetuning on low-resource languages (Conneau, et al., 2020). The authors had publicly released different sizes of the pretrained model checkpoints, out of which the Wav2Vec2-XLS-R-1B (XLSR-1B) [5] model checkpoint with 1B parameters was chosen for finetuning.

Whisper: Unlike Baevski, et al. (2020) which has relied on unsupervised pretraining on large amounts of raw speech, the authors of Whisper (Radford, et al., 2022) took the approach of pretraining a Transformer (Vaswani, et al., 2023) model with semi-supervised learning. The authors collected 680,000 hours of speech audio and corresponding text transcription data of different qualities from web sources for training (Radford, et al., 2022). Apart from just speech transcription, the authors trained the model in a multitask training format to perform a variety of speech processing tasks including, speech translation, language identification and voice activity detection (Radford, et al., 2022). Whisper uses a byte-level Byte Pair Encoding (BPE) text tokenizer and as opposed to typical ASR models, does not normalize the transcription text during training (Radford, et al., 2022). This results in more natural transcription that doesn't require further processing such as punctuation restoration (Radford, et al., 2022). The authors demonstrated that Whisper is able to achieve good performance on its supported languages in a zero-shot setting without any finetuning (Radford, et al., 2022). However, finetuning has been shown to further improve this performance. More interestingly, it has also been shown that Whisper could be finetuned on a new language that was not included in the original training data, by setting the target language to the phonetically closest language among the supported languages. Dhivehi is not officially supported by Whisper, but its closest neighbour Sinhalese is supported, which was used as a target language to finetune the model for Dhivehi. Whisper authors had also released pretrained checkpoints of different sizes, out of which, the Whisper Small [6] checkpoint containing 244M model parameters was chosen for this work.

Massively Multi-lingual Speech (MMS): Whereas XLSR was pretrained on 53 languages, the authors of MMS (Pratap, et al., 2023) scaled this to 1,107 languages. Their primary data source consisted of recordings of people reading translations of the New Testament in different languages (Pratap, et al., 2023). The authors created a GPU accelerated version of the Viterbi algorithm for computing the forced alignment of these recordings to the corresponding texts (Pratap, et al., 2023). Using this forced alignment method, the training dataset was constructed by chunking the recordings and texts into samples of short durations (Pratap, et al., 2023). The authors had used the same model architecture as XLSR

[5] https://huggingface.co/facebook/wav2vec2-xls-r-1b

[6] https://huggingface.co/openai/whisper-small

(Radford, et al., 2022) for MMS (Pratap, et al., 2023). Some of the models they had trained were fully multilingual with all the weights shared across all the training languages, while other models had used language specific adapter layers added to the encoder Transformer blocks (Pratap, et al., 2023). These language adapters constitute an additional 2M parameters which can be swapped out on the fly depending on the language being transcribed. They can also be finetuned separately without finetuning the whole model (Pratap, et al., 2023). The authors recommend finetuning only the language adapters for low-resource languages, but do suggest that full model finetuning is beneficial when more training data is available (Pratap, et al., 2023). The authors had released pretrained model checkpoints of different sizes, out of which MMS-1B-ALL (MMS-1B) [7] model checkpoint with 1B parameters was chosen for finetuning.

3.2 CTC Output Vocabularies

For finetuning the CTC based XLSR and MMS models, an output vocabulary had to be modelled. Previous works (Wav2Vec2-XLS-R-1B-dv) had used a character-based vocabulary of all 49 Thaana symbols and the Arabic ligatures Allah ﷲ (U+FDF2) and *Sallallahou Alayhe Wasallam* (Peace be Upon Him) ﷺ (U+FDFA) which commonly appear in Dhivehi text. However, Xu, et al. (2019) and Zhou, et al. (2021) had shown that training on sub-word based vocabularies where the tokens better correspond to phonemes can produce better results than simple character-based vocabularies. Similarly, Hou, et al. (2020a)'s results also show sub-word vocabularies generally giving better performance. Therefore, it was decided to also train using a more acoustically relevant sub-word vocabulary modelled with all consonants, all consonant-vowel pairs and aforementioned Arabic ligatures. Vowel diacritics were not included separately in this vocabulary as phonetically in Dhivehi, the vowels by themselves do not make any sound. In both vocabularies, additional special tokens were included, which were: the word delimiter token for spaces, the "[UNK]" token for unknown tokens, and the "[PAD]" token for the CTC blank token ϵ. The character-based vocabulary had 54 tokens while the sub-word vocabulary had 461 tokens.

3.3 Speech Dataset

The Dhivehi speech dataset chosen for training the ASR models were taken from Mozilla Common Voice (Ardila, et al., 2019). Common Voice (CV) is a crowd-sourced dataset where volunteers contribute recordings by reading text samples through a web interface (Ardila, et al., 2019). For version 13.0 of Common Voice that was used for this work, there were in total 64 hours of Dhivehi recordings from 331 different speakers. Out of these, only 38 hours were validated by contributors to be correct. The publishers further split these validated hours into different splits, out of which the "train" and "other" splits were selected for model training, "validation" split for evaluation during training and the "test" split for final model evaluation. The average length of samples in the dataset is 4.9 seconds.

Split	Duration	Samples
Train*	25:19:33	19,072
Validation	3:18:39	2,227
Test	3:21:26	2,212
Total	31:59:38	23,511

Table 1: Dataset splits. *For the train split, the original "train" and "other" splits have been combined.

3.4 Text Corpus

To train the language models for CTC LM decoding and for building synthetic spelling correction datasets, a Dhivehi text corpus was needed. For this, a text corpus was built using 4.2GB of text extracted from 1,197,470 news articles provided by ArchiveMV [8], a Maldives online news archive. To build the text corpus, the raw texts were tokenized into sentences using NLTK (Bird, et al., 2009), and the sentences were normalized by removing punctuation and replacing multiple whitespaces with single whitespaces. The text corpus contained 18,117,809 sentences and 258,345,316 tokens of 3,744,110 types.

3.5 Pretrained Text-to-Text Models

For the spelling correction task, a suitable Text-to-Text model architecture had to be chosen. As with

[7] https://huggingface.co/facebook/mms-1b-all

[8] https://archive.mv/

the ASR models, it was decided to use a pretrained model for this task to take advantage of the benefits of transfer learning. The model chosen was the UniMax (Chung, et al., 2023) model by Google Research. This is a multilingual version of the T5 (Raffel, et al., 2020) Text-to-Text Transformer model trained using a novel fairer language sampling method. The authors had released different sized pretrained checkpoints of this model, and the specific checkpoint chosen was the umT5-Small[9] checkpoint containing 300M model parameters.

4 Implementation

4.1 Data Processing

For preprocessing of the speech dataset for training, the text samples were normalized to remove all punctuation marks, which included removing the following characters; ،؟,?.!-;:"‟%‵″�—′…– . Furthermore, any newline characters or multiple spaces were replaced with single spaces. For training the Whisper model, the text was not normalized, but the generated output text and the reference text were normalized when calculating the WER. For the audio data, the audio files were resampled to 16Khz as this was the sampling rate the pretrained models were trained on.

4.2 ASR Model Training

The ASR models were trained using Hugging Face Transformers package (Wolf, et al., 2020), using PyTorch (Paszke, et al., 2019). All the models were trained using the AdamW (Loshchilov & Hutter, 2019) optimizer using a linear learning rate schedule with a warmup of 500 steps. For the MMS and XLSR models, a learning rate of 4.5e-05 was used, while a learning rate of 1e-5 was used for Whisper models. MMS and XLSR models were trained for 30 epochs (except for the MMS-1B-VL-DL-dv which was trained for 40 epochs) while Whisper models were trained for 15 epochs. During training, the models were evaluated every 400 steps using the validation set and a checkpoint saved. At the end of training, the checkpoint with the lowest WER was loaded and saved.

It was noted for MMS and XLSR models using the character-based vocabulary, the training reached the best performance around epoch 2, and beyond this the loss and the WER goes back up. This behaviour was not observed for the models using the sub-word vocabulary, which had a smooth decline of WER until the end of the training.

For both the MMS and XLSR models, the CNN feature extractor layers (Baevski, et al., 2020) were frozen during training. For the MMS models, initial experiments were conducted to train only the language adapter layers (Pratap, et al., 2023), however this resulted in relatively poor performance. So, for the final model training, full model training was performed for the MMS models.

The training was conducted on a AMD Ryzen 9 7950X 16-Core Processor machine with 128GB of RAM and dual RTX 3090 Ti 24GB GPUs running Ubuntu 23.04. Even though the machine had 2 GPUs, training was conducted using single GPUs as there were issues with parallel training the MMS and XLSR models. However, training sessions were conducted two at a time, with sessions assigned to either of the GPUs to take advantage of them both. The XLSR and Whisper models on average took around 11.5 hours to train, while the MMS model trained for 30 epochs took around 13 hours and the model trained for 40 epochs took around 17 hours.

Model Name	Base Model	Vocab.
XLSR-1B-VS-DL-dv	XLSR-1B	character
XLSR-1B-VL-DL-dv	XLSR-1B	sub-word
MMS-1B-VS-DL-dv	MMS-1B	character
MMS-1B-VL-DL-dv	MMS-1B	sub-word
Whisper-Small-DL-dv	Whisper-Small	-

Table 2: ASR models trained.

4.3 Language Models

The language models were trained using KenLM (Heafield, 2011) as 5-gram word-based models. KenLM uses modified Kneser-Ney smoothing[10] and produces efficient inference once trained. The language models were trained using the ArchiveMV text corpus. The first LM was the ArchiveMV-5gram (Amv-5g) model, trained

[9] https://huggingface.co/google/umt5-small

[10] http://www.foldl.me/2014/kneser-ney-smoothing/

without any restrictions. The second LM was the ArchiveMV-5gram-500k (Amv-5g-500k) model which was trained with the vocabulary limited to 500k common Dhivehi words to see whether this had any effect on the performance. For decoding of the CTC ASR model outputs using these LMs, pyctcdecode [11] Python package was used.

4.4 Spelling Correction Models

Model Name	Base Model	Training Samples
umT5-S-1M-dv	umT5-Small	1,043,532
umT5-S-10M-dv	umT5-Small	10,351,970

Table 3: Spelling Correction Models

To train the spelling correction models, first 2 different spelling correction datasets were created which were the 1M and 10M datasets. To create the datasets, random text samples from the ArchiveMV corpus were taken, which were then normalized, and synthetically spelling mistakes introduced into them using a series of random transformations. These transformations were designed to mimic the specific types of errors observed in the ASR model outputs. These include; randomly repeating characters at the end of the string, randomly inserting spaces within words, randomly removing spaces, interchanging phonetically similar letters and vowels (such as ޝ *sheenu* and ށ *shaviyani*), and random insertions, deletions and modifications of characters. These transformed texts were used as the inputs for the 1M and 10M datasets, and the corresponding original normalized texts were used as the labels. The 1M dataset contained around 1 million samples while the 10M dataset contained around 10.3 million samples. For evaluation, a third dataset was created using the MMS-1B-VS-DL-dv model outputs decoded using the ArchiveMV-5gram LM on the validation set of the speech dataset.

The models were trained using the Hugging Face Transformers package, using PyTorch (Paszke, et al., 2019). All the models were trained for only 1 epoch as Transformer models are shown to overfit quickly to training data. AdamW optimizer was used with a learning rate of 4e-5. During training, the model was evaluated using the evaluation dataset and the relative WER improvement was recorded. At the end of the training the checkpoint with the highest relative WER improvement was loaded and saved. The spelling models were also trained using the same machine as the ASR models. The 1M model took around 5 hours to train, while the 10M model took around 56 hours.

5 Evaluation

5.1 Experimental Setup

All the experiments were conducted on the same machine as used for training. Each experiment was conducted on a single GPU using an evaluation script that ran the ASR inference using all the trained ASR models, language models and spelling correction models and recorded the results into CSV files. For the spelling correction evaluations, for the CTC-based ASR models, the outputs from the ArchiveMV-5gram LM decoding was used as a baseline. For the Whisper model, the normalized text outputs were used as inputs to the spelling correction models.

5.2 Baseline ASR Model Results

First, we evaluate the baseline WER of all the trained models on the test sets of all the speech datasets without using any LM decoding or spelling correction and compare the results with the state-of-the-art Dhivehi ASR model Wav2Vec2-XLS-R-1B-dv. Here, for the CTC based XLSR and MMS models, Greedy Search decoding was used. And for the Whisper models, the generated texts were normalized as described in 4.1 before calculating the WER.

Model	Vocabulary	WER
Wav2Vec2-XLS-R-1B-dv	character	20.95
This work		
XLSR-1B-VS-DL-dv	character	56.29
XLSR-1B-VL-DL-dv	sub-word	19.94
MMS-1B-VS-DL-dv	character	38.26
MMS-1B-VL-DL-dv	sub-word	**16.19**
Whisper-Small-DL-dv	-	43.13

Table 4: Baseline ASR Model Results

[11] https://github.com/kensho-technologies/pyctcdecode

As can be seen from the results, our XLSR and MMS based XLSR-1B-VL-DL-dv and MMS-1B-VL-DL-dv models were able to beat the state-of-the-art model on the CV-13 test benchmark. MMS-1B-VL-DL-dv had the best result with a relative improvement of 22.72% as compared to the state-of-the-art. It must be noted that MMS-1B-VL-DL-dv was trained for 10 additional epochs as compared to the other XLSR and MMS models. Therefore, this suggests that the other models also could potentially benefit from longer training.

Interestingly, the best performing CTC based XLSR and MMS models had used the sub-word vocabulary. This is consistent with the results observed by Xu, et al. (2019) and Zhou, et al. (2021) as the sub-words better correspond to phonetic characteristics of Dhivehi, making it easier for the models to learn a relationship between them and the audio. The sub-word vocabulary was able to improve the average WER for both MMS and XLSR models, with an average relative WER improvement of 61.79%. This goes to show that better sub-word modelling using just a little domain knowledge of the target language can go a long way in improving ASR model performance. The current sub-word vocabulary doesn't include all possible Dhivehi phonemes, such as consonant-vowel pairs followed by *sukun* (eg. އުން *un*) or diphones like އައި *a-i.* Further investigations need to be done to see whether expanding the sub-word vocabulary to include these phonemes would improve performance.

5.3 Language Model Decoding Results

Model	Amv-5g-500k	Amv-5g
Wav2Vec2-XLS-R-1B-dv	20.03	19.85
This work		
XLSR-1B-VS-DL-dv	52.63	52.42
XLSR-1B-VL-DL-dv	18.66	18.44
MMS-1B-VS-DL-dv	39.50	38.99
MMS-1B-VL-DL-dv	**14.69**	**14.49**

Table 5: LM decoding results; WER for each of the trained XLSR and MMS models and the state-of-the-art Dhivehi ASR model when decoded using the 2 different LMs. Marked in bold is the best WER obtained for each LM.

The results show a further improvement of WER when LM decoding is used. Using the ArchiveMV-5gram LM, the WER of the MMS-1B-VL-DL-dv model is reduced to 14.49%, which is a 10.50% relative improvement over baseline. The ArchiveMV-5gram LM showed the overall best performance, followed by the ArchiveMV-5gram-500k LM which had the vocabulary limited to 500k words. This indicates that, for this particular case, limiting the LM vocabulary is worse for performance. Table 6 shows that the ArchiveMV-5gram LM improves the WER by 5.65% on average across all models.

Model	Amv-5g-500k	Amv-5g
Wav2Vec2-XLS-R-1B-dv	4.39%	5.25%
This work		
XLSR-1B-VS-DL-dv	6.50%	6.88%
XLSR-1B-VL-DL-dv	6.42%	7.52%
MMS-1B-VS-DL-dv	-3.24%	-1.91%
MMS-1B-VL-DL-dv	**9.26%**	**10.50%**
Average	4.67%	**5.65%**

Table 6: Relative WER improvement for LMs; Showing the relative WER improvement for all XLSR and MMS models using the 2 different LMs as compared to their baseline overall WER without LM decoding.

5.4 Spelling Correction Results

Model	umT5-S-10M-dv	umT5-S-1M-dv
Wav2Vec2-XLS-R-1B-dv	20.83	21.28
This work		
XLSR-1B-VS-DL-dv	50.14	50.73
XLSR-1B-VL-DL-dv	18.17	18.95
MMS-1B-VS-DL-dv	38.52	39.01
MMS-1B-VL-DL-dv	**14.26**	**15.27**
Whisper-Small-DL-dv	42.77	43.54

Table 7: Spelling correction results; WER for each of the trained models & the SOTA Dhivehi ASR model when spelling correction applied. For the MMS and XLSR models, spelling correction was applied after decoding with the ArchiveMV-5gram LM. For the Whisper model, spelling correction was applied after normalizing the output text.

The results show even more WER reduction with spelling correction. Using the umT5-S-10M-dv spelling model, WER for the MMS-1B-VL-DL-dv model is now reduced to 14.26%, which is the new state-of-the-art WER for CV-13 Dhivehi benchmark [12] [13] as of August 2023. This is a 31.93% relative WER improvement as compared to the 20.95% baseline WER of the SOTA Dhivehi ASR model on CV-13.

As can be seen from Table 8, only the umT5-S-10M-dv spelling correction model trained on 10M samples produced any overall WER improvement on average across all the models. This indicates the importance of training on more data. Interestingly, the Whisper model seem to benefit less from spelling correction, which is to be expected as the AED architecture of Whisper essentially learns a language model on the training transcriptions, negating the need for further text processing.

Model	**umT5-S-10M-dv**	**umT5-S-1M-dv**
Wav2Vec2-XLS-R-1B-dv	-4.94%	-7.20%
This work		
XLSR-1B-VS-DL-dv	**4.35%**	**3.22%**
XLSR-1B-VL-DL-dv	1.46%	-2.77%
MMS-1B-VS-DL-dv	1.21%	-0.05%
MMS-1B-VL-DL-dv	1.59%	-5.38%
Whisper-Small-DL-dv	0.83%	-0.95%
Average	**0.75%**	-2.19%

Table 8: Relative WER improvement for spelling correction; Showing the relative WER improvement for all the ASR models using the 2 different spelling correction models as compared to their baseline overall WER without spelling correction.

While spelling correction in addition to LM decoding seems to be a viable technique to boost the accuracy of CTC models, the relative improvement going from LM decoding to spelling correction is markedly less compared to the relative improvement going from baseline to LM decoding. Moreover, doing spelling correction on top of LM decoding adds additional memory and processing time overhead. The memory overhead could be mitigated by loading the spelling model after ASR inference is complete, however, the processing time can still be up to 1.6 times slower than LM decoding alone.

6 Conclusion

Our results show that pretrained multilingual ASR models can greatly benefit from language specific finetuning and optimizations. Specifically for CTC based models, proper sub-word modelling and language model decoding seems crucial. Multilingual models could also benefit from these techniques, as the language optimized sub-word vocabularies can be incorporated back into multilingual models and language specific LMs can be swapped out on the fly when decoding outputs for a specific language. While AED models seem to benefit less from these accuracy boosting techniques, they also seem to benefit from language specific finetuning. As for further accuracy improvement of Dhivehi ASR, more labelled speech data is needed for training, which could be generated using forced alignment as was done by Pratap, et al., 2023. Moreover, further expansion of the CTC sub-word vocabulary can be explored to see if it yields any improvement. Furthermore, for production systems, punctuation restoration Alam, et al., 2020 will need to be performed on the generated text to produce more readable transcripts, especially for the CTC based models.

Acknowledgments

I would like to thank my supervisor Dr. Mark-Jan Nederhof for his invaluable insight and guiding me throughout the project. A special thanks to the Commonwealth Scholarship Commission (CSC) and the UK Foreign, Commonwealth & Development Office (FCDO) and the British people for giving me the opportunity to study at the University of St. Andrews. Special thanks to Naail Abdul Rahman (Kudanai) for his valuable insight into Dhivehi ASR and sharing his past results with me. A similar thanks to Shahuza Abdul Kareem (Shahu) for her ground-breaking works on Dhivehi ASR and sharing her results with me. Thanks to Afrah Ismail for giving access to the ArchiveMV data. A very special thanks to CEO of Javaabu

[12] https://paperswithcode.com/sota/speech-recognition-on-common-voice-dhivehi

[13] https://huggingface.co/shahukareem/wav2vec2-xls-r-1b-dv

(where I'm a Co-founder) Mohamed Jailam for providing the hardware for training and evaluation without which this work would not have been possible. Finally, a special thanks to my lovely wife, Iju for always being there for me.

References

Tanvirul Alam, Akib Khan, and Firoj Alam. 2020. Punctuation Restoration using Transformer Models for High-and Low-Resource Languages. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 132–142, Online. Association for Computational Linguistics.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Gregor Weber. 2019. Common voice: A massively-multilingual speech corpus. *arXiv preprint arXiv:1912.06670. Verson 2.*

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *arXiv preprint arXiv:2006.11477. Version 3.*

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473. Version 7.*

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* O'Reilly Media, Inc.

Hyung Won Chung, Noah Constant, Xavier Garcia, Adam Roberts, Yi Tay, Sharan Narang, and Orhan Firat. 2023. UniMax: Fairer and more Effective Language Sampling for Large-Scale Multilingual Pretraining. *arXiv preprint arXiv:2304.09151. Version 1*

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. 2020. Unsupervised Cross-lingual Representation Learning for Speech Recognition. *arXiv preprint arXiv:2006.13979. Version 2.*

Alex Graves, Santiago Fernández, Faustino John Gomez, and Jürgen A. Schmidhuber. 2006. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 369–376. New York, NY, USA. Association for Computing Machinery.

Awni Hannun. 2017. Sequence Modeling with CTC. *Distill.*

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. 2014. Deep Speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567. Version 2.*

Ibrahim Hassaan, Ifham Mohamed, Raaid Adam Rasheed, and Yameen Mohamed. 2018. Dhivehi automatic speech recognition system. *Project, Faculty of Engineering Science and Technology, Maldives National University.*

Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197. Edinburgh. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-term Memory. *Neural computation, Volume 9, Issue 8*, pages 1735-1780.

Wenxin Hou, Yue Dong, Bairong Zhuang, Longfei Yang, Jiatong Shi, and Takahiro Shinozaki. 2020a. Large-scale end-to-end multilingual speech recognition and language identification with multi-task learning. In *Proceedings of Interspeech 2020,* pages 1037-1041. Shanghai, China.

Wenxin Hou, Yue Dong, Bairong Zhuang, Longfei Yang, Jiatong Shi, and Takahiro Shinozaki. 2020b. Super Multi-lingual End-to-End Speech Recognition and Its Transfer learning to Low Resource Languages. *IPSJ SIG Technical Report, Vol.2020-SLP-133 No.8*

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101. Version 3*

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach 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.

Ngoc-Quan Pham, Tuan-Nam Nguyen, Sebastian Stueker, and Alexander Waibel. 2021. Efficient weight factorization for multilingual speech recognition. *arXiv preprint arXiv:2105.03010. Version 1.*

Rohit Prabhavalkar, Takaaki Hori, Tara N. Sainath, Ralf Schlüter, and Shinji Watanabe. 2023. End-to-End Speech Recognition: A Survey. *arXiv preprint arXiv:2303.03329. Version 1.*

Vineel Pratap, Andros Tjandra, Bowen Shi, Paden Tomasello, Arun Babu, Sayani Kundu, Ali Elkahky, Zhaoheng Ni, Apoorv Vyas, Maryam Fazel-Zarandi, Alexei Baevski, Yossi Adi, Xiaohui Zhang, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. 2023. Scaling Speech Technology to 1,000+ Languages. *arXiv preprint arXiv:2305.13516. Version 1.*

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever 2022. Robust Speech Recognition via Large-Scale Weak Supervision. *arXiv preprint arXiv:2212.04356. Version 1.*

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. *arXiv preprint arXiv:1910.10683. Version 4.*

Francis M. Tyers and Josh Meyer. 2021. What shall we do with an hour of data? Speech recognition for the un-and under-served languages of Common Voice. *arXiv preprint arXiv:2105.04674. Version 1.*

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. *arXiv preprint arXiv:1706.03762. Version 7.*

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.

Hainan Xu, Shuoyang Ding, and Shinji Watanabe. 2019. Improving End-to-end Speech Recognition with Pronunciation-assisted Sub-word Modeling. *arXiv preprint arXiv:1811.04284. Version 2.*

Shiliang Zhang, Ming Lei, and Zhijie Yan. 2019. Automatic Spelling Correction with Transformer for CTC-based End-to-End Speech Recognition. *arXiv preprint arXiv:1904.10045. Version 1.*

Yu Zhang, Wei Han, James Qin, Yongqiang Wang, Ankur Bapna, Zhehuai Chen, Nanxin Chen, Bo Li, Vera Axelrod, Gary Wang, Zhong Meng, Ke Hu, Andrew Rosenberg, Rohit Prabhavalkar, Daniel S. Park, Parisa Haghani, Jason Riesa, Ginger Perng, Hagen Soltau, Trevor Strohman, Bhuvana Ramabhadran, Tara Sainath, Pedro Moreno, Chung-Cheng Chiu, Johan Schalkwyk, Françoise Beaufays, and Yonghui Wu. 2023. Google USM: Scaling Automatic Speech Recognition Beyond 100 Languages. *arXiv preprint arXiv:2303.01037. Version 3.*

Wei Zhou, Mohammad Zeineldeen, Zuoyun Zheng, Ralf Schlüter, and Hermann Ney. 2021. Acoustic Data-Driven Subword Modeling for End-to-End Speech Recognition. In *Proceedings of Interspeech 2021*, pages 2886-2890. Brno, Czech Republic.

Comparing Modular and End-To-End Approaches in ASR for Well-Resourced and Low-Resourced Languages

Aditya Parikh Louis ten Bosch Henk van den Heuvel Cristian Tejedor-García
Centre for Language and Speech Technology,
Radboud University, Nijmegen, The Netherlands
{aditya.parikh,louis.tenbosch,henk.vandenheuvel,cristian.tejedorgarcia}@ru.nl

Abstract

We present a comparative study of a state-of-the-art traditional modular Automatic Speech Recognition (Kaldi ASR) and an end-to-end ASR (wav2vec 2.0) for a well-resourced language (Spanish) and a low-resourced language (Irish). We created ASRs for both languages and evaluated their performance under different update regimes. Our results show that the end-to-end wav2vec 2.0 outperforms the modular ASR for both languages in terms of Word Error Rate (WER) but performs worst in terms of real-time decoding. We also addressed the issue of non-lexical words in wav2vec 2.0's output. We found that in wav2vec 2.0 by LM integration with shallow fusion and increasing LM weight to 0.7 and 0.8 respectively for the Spanish and Irish provided the optimum ASR performance by reducing non-lexical words. However, this does not eliminate all non-lexical words. Finally, our study found that Kaldi ASR would perform best for real-time decoding for longer audio inputs compared to wav2vec 2.0 model trained on the same dataset on the minimal infrastructure, although wav2vec 2.0's performance can be improved with a GPU acceleration in backend. These results may have significant implications for creating real-time ASR services, especially for low-resourced languages.

1 Introduction

Traditional modular ASR frameworks decompose the ASR task into acoustic, pronunciation, and language modeling e.g. Povey et al. (2011). The modular approach of ASR is knowledge-based and provides flexibility in training one's own acoustic model (AM) and language model (LM), in combination with a dedicated customised vocabulary. The knowledge-based modular approach allows adequate performance in specific domains like specific languages, dialects or speakers. A modular ASR can be tailored to the specific domain or task, which can lead to further improvement of the performance of the system (Roy et al., EasyChair, 2022). However, the traditional modular approach of ASR requires a significant amount of transcribed speech recording for training, large text resources, and explicit grapheme-to-phoneme (G2P) mappings or complete dictionaries as basic requirements. This poses a significant challenge for low-resourced languages that do not have a significant digital footprint with a limited amount of labeled data available (Srivastava et al., 2018).

Self-supervised learning (SSL) has emerged as a powerful technique for settings where annotated audio data is scarce. The key idea behind this approach is to learn (pretrained) general representations from substantial amounts of unlabeled source data, and subsequently leverage them to improve the performance (finetuning) on downstream target tasks with a very limited amount of transcribed data. This is particularly useful for tasks such as speech recognition, where obtaining labeled data can be a time-consuming and costly process. Models based on SSL, e.g. wav2vec 2.0 (Baevski et al., 2020), have shown their powerful representation ability and feasibility for ultra-low-resourced speech recognition, making self-supervised end-to-end models a desirable alternative to the flexible and useful modular infrastructure.

This paper aims to evaluate and compare the performance of two different approaches for developing ASR systems: modular Kaldi ASR e.g., Povey et al., 2011 and end-to-end ASR based on wav2vec 2.0 (Baevski et al., 2020), for two languages: Spanish (well-resourced) and Irish (low-resourced). The study not only assesses the performances of both approaches in terms of WER but also addresses challenges with wav2vec 2.0 such as generating non-lexical word forms (such as 'weekent', 'halloo') and the impact of LM weights. Additionally, we examine the latencies and real-time factor (RTF) while deploying both ASRs under the same client-server network environment. In this way, we aim to determine which approach is more effective for developing ASR systems for different languages and resource levels, specifically with minimal infrastructure.

2 Related Work

Since the emergence of self-supervised learning methods, various studies showcased the potential of self-supervised end-to-end approaches in speech technology across different languages and modalities (Zuluaga-Gomez et al., 2023; Coto-Solano et al., 2022; Al-Ghezi et al., 2021; Yi et al., 2021). One such study by Zuluaga-Gomez et al. (2023) examines the robustness of two end-to-end models wav2vec 2.0 and XLS-R trained in a new domain, air traffic control (ATC) communications. Their findings show significant reductions in relative WER ranging from 20% to 40% compared to the hybrid-based ASR baseline, indicating the effectiveness of self-supervised end-to-end approaches in this domain. Another study by Coto-Solano et al. (2022) was conducted on Cook Islands Maori (CIM), a low-resourced indigenous language, to compare the performance of three ASR models: A traditional modular system (Kaldi; Povey et al., 2011) and two deep learning-based systems (DeepSpeech (Hannun et al., 2014) and XLSR-wav2vec 2.0 (Conneau et al., 2021)) and their results also indicated that Deep Learning ASR systems XLSR-wav2vec 2.0 are performing at the level of modular ASR methods on small datasets, and they are also effective in dealing with extremely low-resourced Indigenous languages like CIM. A study on Swedish L2 learners by Al-Ghezi et al. (2021) found that models pre-trained on large size of untranscribed L1 Swedish speech data give a competitive performance to that of modular ASR without the need for customized language and pronunciation models. Their best model managed to correctly decode words that do not appear in the training dataset whereas the modular ASR failed to do so. In Enzell (2022), domain adaptation with an N-gram LM is shown for Swedish, where the effects of LM weights on end-to-end models are briefly discussed.

3 Data

In our research, we utilized various open-source datasets and public speech corpora. For the Spanish ASR, we utilized the Common Voice (CV) Spanish (Ardila et al., 2020) dataset for the AM. The CV dataset includes rich metadata such as speaker age, accent, and gender, and consists of 213244 utterances for training, equating to 313.56 hours of speech material. For building the LM, we utilized the Spanish Billion Words Corpus (Cardellino, 2019) which has nearly 1.5 billion Spanish tokens and 0.54 million types with a frequency higher than 10. For testing, we used the CV Spanish Dev and Test sets, which consist of 26.1 and 25.9 hours of speech, respectively. For pronunciation lexicons we used a dedicated G2P tool based on SAMPA (Speech Assessment Methods Phonetic Alphabet) (Wells et al., 1997). It's worth noting that obtaining datasets for Spanish was relatively easy as it is a well-resourced language with a substantial digital footprint. See the Table 1.

Table 1: Overview of Common voice Spanish Datasets

Dataset	#Utterances	Duration	#Word Token	#Word Type
Train	213244	313.56h	2124011	83604
Test	15440	26.1h	151681	23314
Dev	15440	25.9h	151819	23602

For Irish, the situation is essentially different. Acquiring speech data for this language is a significant challenge due to the scarcity of open-source resources available for this language. To tackle this scarcity problem, we combined multiple small open-source Irish datasets. For the AM training we utilized the CV Irish dataset (Ardila et al., 2020). We used only the validated utterances from this dataset and excluded those that were part of the test set. Additionally, we used the "Living Audio" dataset (Braude et al., 2019) which contributed an additional hour of Irish speech data. We also incorporated all Irish utterances from the "Google Fleurs" dataset (Conneau et al., 2023). After combining these three datasets, we were able to train on a total of 9,274 utterances equating to 13.5 hours of speech. For testing, we used the CV Irish Test set, containing 513 utterances (0.5 hours of speech), in combination with a set of 'Invalidated' CV Irish utterances, with 282 utterances (0.3 hours of speech, after removing speech samples with background noise or no speech). The 'invalidated' clips in the CV dataset are the clips that have received more downvotes than upvotes. In Table 2, the overview of Irish datasets is provided.

For the LM, we used the CC-100: Monolingual datasets from Web Crawl Data (Conneau et al., 2020), which includes data for over 100 languages including Irish, with in total 84 million word tokens and 0.12 million word types having frequency higher than 10. Lastly, for permitting the experiments with Kaldi ASR, we trained a G2P model based on Joint-sequence models (Bisani and Ney, 2008) using 13300 seed Irish pronunciations acquired from Wikipron (Lee et al., 2020).

Table 2: Overview of Irish Datasets. **CV**, **LA** and **GF** abbreviated for Common Voice, Living Audio and Google Fleurs respectively.

Dataset	**#Utterances**	**Duration**	**#Word Tokens**	**#Word Types**
CV Train	4097	4.1h	27880	2341
LA Irish	1122	1h	11360	3542
GF Irish	1947	8.4h	48929	9866
CV Test	513	0.5h	3423	1109
CV Invalidated	282	0.3h	2230	707

4 Experiments

Our experiment setup is composed of four objectives: 1. Evaluate the performance of both the modular and end-to-end ASR approaches in terms of WER and Character Error Rate (CER), 2. Examine the influence of LM weights when integrating with fine-tuned wav2vec 2.0 models 3. Evaluate the presence of non-lexical words in the generated transcriptions and 4. Measure the latency of the ASR systems when deployed using both methods.

4.1 Modular ASR Training

We established the first baseline modular ASR for Spanish and Irish languages, using the dataset specified in Section 3. For the Spanish ASR, the baseline was built using a Kaldi (Povey et al., 2011) chain model adapted from the Librispeech recipe[1], while for the Irish ASR, it was adapted from the mini-Librispeech[2] recipe. Both recipes follow a similar training pattern, but the hyperparameters such as the number of leaves, number of Gaussians, neural network size, L2 regularization, learning rate, and the number of epochs were optimized to fit the data size. The AM in both recipes is a combination of a Time-Delayed Neural Network (TDNN) and a Convolutional Neural Network (CNN). Additionally, 4-gram statistical LMs for Spanish and Irish were generated using the SRILM tool (Stolcke, 2002), based on the text resources for these languages. Finally, the pronunciation lexicons were created using a rule-based approach for Spanish and a data-driven approach for Irish as explain in the section 3.

4.2 Finetuning with End-to-End Approach

We utilized a publicly released pre-trained wav2vec 2.0 model (Baevski et al., 2020), XLS-R (Babu et al., 2022), which was trained on 436K hours of publicly available speech audio and is available on HuggingFace[3]. During its self-supervised pre-training, XLS-R learned contextualized speech representations by randomly masking feature vectors and passing them through a transformer network. For fine-tuning on our speech recognition task, we added a single linear layer on top of the pre-trained network and finetuned the model on our labeled speech data for both Spanish and Irish. We used the 300 million-parameter version of XLS-R[4], which is among the smaller versions (mid 2023, models range from 300 million to two billion parameters). The fine-tuning was performed on an NVIDIA Tesla T4 GPU using the Adam optimizer, with a learning rate starting with a warm-up for 500 steps, peaked at $3e^{-4}$ for all global steps, and then decayed exponentially. The total number of global steps for fine-tuning to Spanish and Irish was 44415 and 7180, respectively. In our research, the same language-dependent statistical LM was used for the modular and on end-to-end approach, for both Spanish and Irish. These LMs were initially created in ARPA format but were transformed into binary using KENLM (Heafield et al., 2013) to decrease the time required to load the models. The integration of the LM with the AM was performed using shallow fusion through the CTC decoder library `pyctcdecode`[5].

4.3 Non-lexical Words

An ASR system based on CTC may produce non-lexical word forms. In wav2vec 2.0, the output of the model is represented as the probability distribution of the predicted phonemes at each time frame (each 20ms) of the input signal. While the model can generate both lexical and non-lexical word forms through its sequence of phonemes, the use of a (word-based) LM helps to refine non-lexical predictions by incorporating information about the likelihood of different sequences of phonemes that form words (legal grapheme sequences or legal phone sequences) in the language.

[1] github.com/kaldi-asr/kaldi/blob/master/egs/librispeech/

[2] github.com/kaldi-asr/kaldi/blob/master/egs/mini_librispeech/

[3] huggingface.co/docs/transformers/model_doc/wav2vec2

[4] huggingface.co/facebook/wav2vec2-xls-r-300m

[5] github.com/kensho-technologies/pyctcdecode/

In contrast, Kaldi's lexicon search space is limited to the pronunciation lexicons. The HCLG graph in Kaldi uses the lexicon FST (Povey et al., 2011; Mohri et al., 2007) to determine the possible words based on the AM's predictions, effectively restricting the search space to the words defined in the lexicon. This ensures that Kaldi only produces words that we provide, rather than generating non-existing words, leading to more accurate results.

In wav2vec 2.0, we investigated the effect of varying the weight of the LM during the shallow fusion process, by calculating the number of unique words (word types) in each experiment for different values of LM weights ranging from 0 to 1, with intermediate values of 0.1, 0.3, 0.5, 0.7, 0.8, 0.9, and 1.0. The results of these experiments allowed us to observe the effects of non-lexical words in hypothesis transcripts generated by wav2vec 2.0.

4.4 ASR Usability in Deployment

The ASR created using both approaches was deployed as a web service. The Kaldi-based ASR pipeline is capable of processing most speech files faster than real-time using only CPUs (Parikh et al., 2022).

However, decoding with large wav2vec 2.0 models with an integrated LM is prohibitively slow on a CPU and therefore requires the availability of at least one GPU for real-time decoding. Additionally, the wav2vec 2.0 models needed to be manually loaded for the first time setup. The latency of the ASR web service is an important feature for the usability of the entire system and user satisfaction. We calculated the latency results in terms of RTF for audio files of varying durations for both Kaldi ASR and wav2vec 2.0 models while maintaining a consistent connection environment. Linear regression was used to obtain equations. The linear trendlines were obtained by fitting linear models to each dataset using the Ordinary Least Squares (OLS) method. The slope and intercept coefficients of each line were calculated using the linear regression model.

5 Results

The initial evaluation of both systems is based on WER. For the modular approach, we conducted online decoding with Kaldi ASR, and for the end-to-end wav2vec 2.0 approach, we computed the WERs for the finetuned model and for the shallow-fused model with various weights of LM.

Table 3: Experimental Results of Kaldi ASR using a CNN-TDNN Architecture for AM: Testing Datasets and Corresponding WER and CER

Spanish ASR		
Dataset	**WER**	**CER**
CV Test	15.69%	5.89%
CV Dev	13.68%	4.90%
Irish ASR		
CV Test	22.69%	11.54%
CV Invalidated	43.06%	24.44%

As shown in Table 3 and 4, the end-to-end wav2vec 2.0 method outperformed the modular Kaldi ASR approach. In Spanish ASR, with Kaldi, we obtained WERs of 15.69% and 13.68% on the CV Test and CV Dev sets, respectively, which were improved to 10.63% and 9.38% by wav2vec 2.0 without an LM. Similarly, in Irish ASR, we obtained WERs of 19.98% and 39.19% using the wav2vec 2.0 without an LM on the CV Test and Invalidated sets, compared to the Kaldi ASR with WERs of 22.69% and 43.06%.

We also determined the impact of an LM on the finetuned model with wav2vec 2.0. As described in section 4.3, we computed the WER and CER for a number of LM weight values. For the Spanish ASR, the lowest WER of 6.73% and 5.92% on the CV Test and CV Dev sets, respectively, was achieved with an LM weight of 0.7. In the Irish ASR, the lowest WER was obtained at an LM weight of 0.8, with WERs of 13.78% and 30.85% on the CV Test and CV Invalidated sets, respectively. These results demonstrate a significant improvement in WER compared to the baseline modular ASR, using the same training data.

We evaluated the impact of the LM weight on the non-lexical words in the hypothesis transcripts generated by Spanish and Irish wav2vec 2.0 models. The non-lexical words were defined as words that were not present in the unigrams of the LM shallow-fused with the wav2vec 2.0 model. As seen in Table 5 initially, without using an LM, there were 6220 and 5770 non-lexical words in the Spanish CV Test and Dev hypothesis transcripts, respectively. By integrating an LM and increasing the weight of LM to 0.5, the non-lexical words were reduced to a minimum of 1317 and 1235 in CV Test and Dev transcripts, respectively corresponding to a reduction of approximately 79% of the total non-lexical words. Similarly in Irish ASR, without using an

Table 4: WER and CER of two test sets for Spanish and Irish ASR by wav2vec 2.0. We recorded WER and CER for fine-tuned model integrated with different LM weights.

Dataset	**Evaluation Matrix**	**No LM**	**LM Weights**								
			0	**0.1**	**0.3**	**0.5**	**0.7**	**0.8**	**0.9**	**1**
Spanish ASR											
CV Test	**WER**	10.63%	10.44%	9.03%	7.43%	6.85%	**6.73%**	6.82%	6.98%	7.20%
	CER	3.09%	2.95%	2.73%	2.44%	2.34%	**2.35%**	2.39%	2.44%	2.49%
CV Dev	**WER**	9.38%	9.06%	7.86%	6.53%	6.03%	**5.92%**	6.01%	6.15%	6.39%
	CER	2.59%	2.47%	2.28%	2.03%	1.94%	**1.93%**	1.97%	2.01%	2.06%
Irish ASR											
CV Test	**WER**	19.98%	19.07%	17.23%	14.95%	13.96%	13.87%	**13.78%**	13.78%	13.87%
	CER	7.24%	6.91%	6.52%	6.03%	5.79%	5.84%	**5.85%**	5.88%	5.89%
CV Invalidated	**WER**	39.19%	39.95%	37.62%	33.45%	31.88%	31.07%	**30.85%**	31.07%	31.39%
	CER	16.81%	16.54%	16.11%	15.39%	15.16%	15.20%	**15.15%**	15.23%	15.28%

Table 5: Count of non-lexical words in transcripts generated by wav2vec 2.0

LM Weight	**Test Dataset**			
	Spanish		**Irish**	
	CV Test	**CV Dev**	**CV Test**	**CV Invalidated**
No LM	6220	5770	339	357
0	3408	3046	257	238
0.1	2440	2184	207	197
0.3	1538	1404	149	150
0.5	1317	1235	124	125
0.7	1404	1324	119	122
0.8	1555	1438	122	128
0.9	1769	1622	124	132
1	1999	1820	127	141

LM, in CV Test and Invalidated, there were 339 and 357 non-lexical words which were reduced to 119 and 122 using an LM with 0.7 weight corresponding to a reduction of approximately 65% of the total non-lexical words. Although the optimal WER and CER were achieved with only marginal differences at LM weights of 0.7 for Spanish and 0.8 for Irish, it can be said that there is still a presence of a small number of non-lexical homophones in hypothesis transcripts. However, even with a high LM weight, not all non-lexical words were removed. A slight increase in the number of non-lexical words was observed as the weight of the LM was increased from 0.7. This highlights the fact that even with low CER produced by wav2vec 2.0 models, there can still be a significant number of non-lexical words present in the generated transcripts.

In Figure 1, we present a comparative analysis of latency times and Real-Time Factors (RTF) for two ASR systems, Kaldi and wav2vec 2.0. This analysis covers audio files ranging from 5 to 102 seconds in duration, all processed under identical testing conditions, including network settings and beam size. Additionally, we consider a scenario where the wav2vec 2.0 model is utilized with a NVIDIA Tesla T4 GPU with 15.36GB of memory. The key observation is that both Kaldi and wav2vec 2.0 exhibit linearly increasing latency times as audio duration extends. For Kaldi, the latency equation is given by $y = 0.1074x + 0.50190$ (y: latency; x: duration in seconds), while for wav2vec 2.0 on the identical testing condition as Kaldi ASR, it is $y = 0.2380x - 0.8749$. When using wav2vec 2.0 with GPU backend, the latency equation becomes $y = 0.0426x + 0.8357$. These equations describes how the latency of ASR system increases as the duration of the audio input increases. In summary, the latency is same in all the ASRs for audio utterances up to 10 seconds. It is also evident that all three systems experience increased latency with longer audio segments. Wav2vec 2.0 displays a higher linear increase, while Kaldi exhibits a slower rate of increase, indicating greater efficiency with longer audio. Notably, wav2vec 2.0 with GPU acceleration demonstrates significantly reduced latency, underscoring the advantages of GPU processing for longer audio tasks. These insights are invaluable for selecting the ideal system for real-time or near-real-time audio processing, considering expected processing times based on varying audio durations.

The RTF values for both systems show an inverse relationship with audio file length. For Kaldi, the estimated RTF is $y = -0.0008x + 0.1791$ (y: RTF; x: duration in seconds). Kaldi's performance is only 0.0161 times better (RTF = 0.129 for 20 seconds of audio to RTF = 0.113 for 102 seconds of audio) for files from 20 to 102 seconds long. In contrast, wav2vec 2.0 model when same system as Kaldi ASR in backend gives an RTF estimated

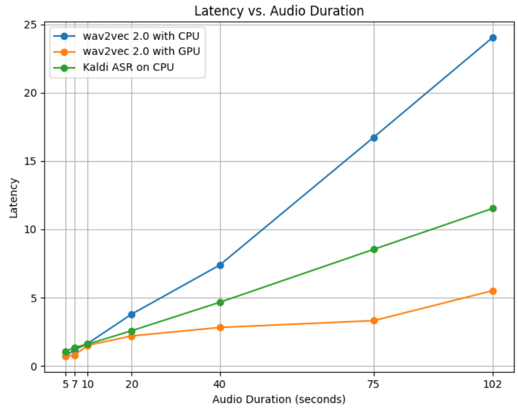


Figure 1: Latency measured in terms of system time for wav2vec 2.0 models and Kaldi vs. Audio Duration

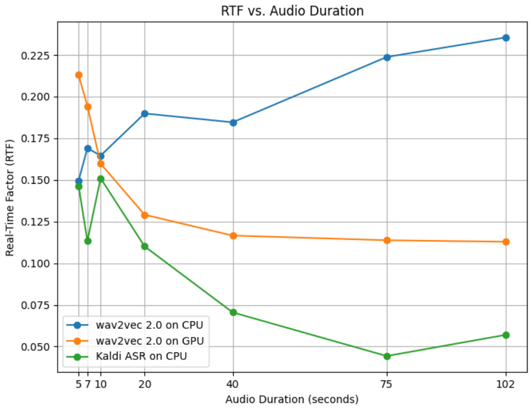


Figure 2: Real-time Factors (system time/length of audio) for wav2vec 2.0 models and Kaldi vs. Audio Duration

by $y = 0.00079x + 0.1588$. In this case, the RTF value is *increasing* with the audio duration, i.e. the system's processing time becomes relatively slower as the audio duration becomes longer. This suggests that the system might not be able to keep up with the real-time demands of longer audio segments, and it could experience prohibitive delays in processing or decoding longer audio. While this issue can be solved with an acceleration at backend as GPU and with GPU, wav2vec 2.0's performance is 3.5 times better for files from 20 to 102 seconds long, with an RTF of $y = -0.0009x + 0.1351$. The initial loading time of the wav2vec 2.0 model (which takes around 10 to 20 seconds) is not taken into consideration in the charts.

6 Discussion

From our experimental results, it is evident that the end-to-end wav2vec 2.0 approach outperforms the modular Kaldi ASR for both well-resourced and low-resourced languages. In particular, we found that in end-to-end wav2vec 2.0 during shallow fusion increasing the LM weight from 0.0 to 0.7 and 0.8 for Spanish and Irish, respectively, led to a decrease in non-lexical words, WER, and CER, resulting in optimum performance. Interestingly, beyond a certain threshold, further increasing an LM weight led to an *increase* in non-lexical words, WER, and a decrease in performance. The wav2vec 2.0 model outputs a sequence of token probabilities represented in an alphabet set and an arg-max followed by a tokenizer provides sufficiently good accuracy but when an LM is integrated on top of it, words with lower probability and poor acoustic support are more likely to be overruled by the LM. Hence a reduction in WER and non-lexical words is found but after a certain limit for the LM-weight, the LM starts replacing correctly identified words resulting in an increase in WER. The default weight of LM in `pyctcdecode` is 0.5, but finding the optimum weight for the combination of LM and AM is crucial for achieving the best performance. In the modular ASR, after the decoding process performing lattice rescoring with recurrent neural LMs (Xu et al., 2018) can also further improve the ASR performance.

The knowledge-based hybrid system and end-to-end systems that we have compared here differ in terms of WER. This does not at all imply that the classical approach can defaultly be replaced by an SSL end-to-end approach. In the experiments reported on above, we observed that both systems often (but not always) make *different* errors, which opens the possibility to consider them as first-level audio-to-text transformations after which both 'streams' could be merged on a second level, based on considering confidence measures associated to each word in the hypothesized outputs in each stream. This stream-based merging of multiple different hypotheses is topic for a follow-up investigation.

In terms of time performance, the fitted latency and RTF lines are reliable indicators of trends in the data and can be used for predictions and insights. For real-time decoding, wav2vec 2.0 takes considerably more time to decode the longer than 10 seconds audio, compare to Kaldi ASR in the same network connection and server infrastructure but with a GPU acceleration, wav2vec 2.0 decoding times outperform Kaldi in all cases, but a first model loading time must be taken into account in the case of wav2vec 2.0.

7 Conclusion

We compared the performance of modular and end-to-end approaches for creating ASR on a low and well-resourced language and results showed that the end-to-end wav2vec 2.0 ASR outperforms the modular Kaldi ASR even without an LM. Incorporating an LM with weights of 0.7 and 0.8 for Spanish and Irish languages, respectively, further improves the performance of the end-to-end approach. However, we observed that the end-to-end approach generates non-lexical words, which can be partially resolved but not entirely eliminated by integrating an LM. Also, a dedicated GPU is required to achieve the best time performance for end-to-end ASR, which is 3.5 times faster than modular ASR. Therefore, modular ASR can still be a relevant option for in-domain tasks with lower CPU/GPU requirements.

Limitations

There are mainly three limitations with our study. 1. The main limitation of this study concerns the data preparation phase, especially for low-resource languages. Conducting experiments, as presented in this paper, requires adequate linguistic resources. It includes not only audio material but also essential components such as lexicons or a grapheme to phoneme conversion system. The scarcity of such linguistic resources for minority languages can pose a significant challenge so the availability of such an ASR system remains crucial for this comparison. 2. Another significant limitation relates to the availability of suitable large models, such as Whisper, for the purpose of comparison. Not all pre-trained end-to-end ASR systems encompass support for every minority or low-resourced language. So the availability of such an ASR system remains crucial for this comparison. 3. Third limitation is the hardware dependent performance. In our case, AMD 32-Core Processor with a total of 64 CPUs, which is also quite capable. However, the performance of ASR systems can be impacted by factors at server side such as CPU load, available memory, and system usage by other processes and at network side such as bandwidth, processing speed, and transmission protocol. This variability can affect the latency and RTF of the ASR system, meaning that the time it takes to process and transcribe speech can vary under different system conditions.

Acknowledgements

This work has been conducted in the SignON project, funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017255.

References

Ragheb Al-Ghezi, Yaroslav Getman, Aku Rouhe, Raili Hildén, and Mikko Kurimo. 2021. Self-Supervised End-to-End ASR for Low Resource L2 Swedish. In *Proc. Interspeech 2021*, pages 1429–1433.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France. European Language Resources Association.

Arun Babu, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. 2022. XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale. In *Proc. Interspeech 2022*, pages 2278–2282.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc.

Maximilian Bisani and Hermann Ney. 2008. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451.

David A. Braude, Matthew P. Aylett, Caoimhín Laoide-Kemp, Simone Ashby, Kristen M. Scott, Brian Ó Raghallaigh, Anna Braudo, Alex Brouwer, and Adriana Stan. 2019. All Together Now: The Living Audio Dataset. In *Proc. Interspeech 2019*, pages 1521–1525.

Cristian Cardellino. 2019. Spanish Billion Words Corpus and Embeddings.

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. 2021. Unsupervised Cross-Lingual Representation Learning for Speech Recognition. In *Proc. Interspeech 2021*, pages 2426–2430.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco

Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Min Ma, Simran Khanuja, Yu Zhang, Vera Axelrod, Siddharth Dalmia, Jason Riesa, Clara Rivera, and Ankur Bapna. 2023. Fleurs: Few-shot learning evaluation of universal representations of speech. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 798–805.

Rolando Coto-Solano, Sally Akevai Nicholas, Samiha Datta, Victoria Quint, Piripi Wills, Emma Ngakuravaru Powell, Liam Koka'ua, Syed Tanveer, and Isaac Feldman. 2022. Development of automatic speech recognition for the documentation of Cook Islands Māori. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 3872–3882, Marseille, France. European Language Resources Association.

Viktor Enzell. 2022. Domain adaptation with n-gram language models for swedish automatic speech recognition : Using text data augmentation to create domain-specific n-gram models for a swedish open-source wav2vec 2.0 model. Master's thesis, KTH, School of Electrical Engineering and Computer Science (EECS).

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. 2014. Deep speech: Scaling up end-to-end speech recognition.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 690–696, Sofia, Bulgaria. Association for Computational Linguistics.

Jackson L. Lee, Lucas F.E. Ashby, M. Elizabeth Garza, Yeonju Lee-Sikka, Sean Miller, Alan Wong, Arya D. McCarthy, and Kyle Gorman. 2020. Massively multilingual pronunciation modeling with WikiPron. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4223–4228, Marseille, France. European Language Resources Association.

Mehryar Mohri, Fernando C. N. Pereira, and Michael Riley. 2007. Speech Recognition with Weighted Finite-State Transducers. In *Handbook on Speech Processing and Speech Communication, Part E: Speech recognition*.

Aditya Parikh, Louis ten Bosch, Henk van den Heuvel, and Cristian Tejedor-García. 2022. Design principles of an automatic speech recognition functionality in a user-centric signed and spoken language translation system. *Computational Linguistics in the Netherlands Journal*, 12:19–32.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, CONF. IEEE Signal Processing Society.

Mukund K Roy, Sunita Arora, Karunesh Arora, and Shyam S Agarwal. EasyChair, 2022. Building speech corpus in rapid manner to adapt a general purpose asr system to specific domain. EasyChair Preprint no. 7181.

Brij Mohan Lal Srivastava, Sunayana Sitaram, Rupesh Kumar Mehta, Krishna Doss Mohan, Pallavi Matani, Sandeepkumar Satpal, Kalika Bali, Radhakrishnan Srikanth, and Niranjan Nayak. 2018. Interspeech 2018 Low Resource Automatic Speech Recognition Challenge for Indian Languages. In *Proc. 6th Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU 2018)*, pages 11–14.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proc. 7th International Conference on Spoken Language Processing (ICSLP 2002)*, pages 901–904.

John C Wells et al. 1997. Sampa computer readable phonetic alphabet. *Handbook of standards and resources for spoken language systems*, 4:684–732.

Hainan Xu, Tongfei Chen, Dongji Gao, Yiming Wang, Ke Li, Nagendra Goel, Yishay Carmiel, Daniel Povey, and Sanjeev Khudanpur. 2018. A pruned rnnlm lattice-rescoring algorithm for automatic speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5929–5933.

Cheng Yi, Jianzong Wang, Ning Cheng, Shiyu Zhou, and Bo Xu. 2021. Transfer ability of monolingual wav2vec2.0 for low-resource speech recognition. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6.

Juan Zuluaga-Gomez, Amrutha Prasad, Iuliia Nigmatulina, Seyyed Saeed Sarfjoo, Petr Motlicek, Matthias Kleinert, Hartmut Helmke, Oliver Ohneiser, and Qingran Zhan. 2023. How does pre-trained wav2vec 2.0 perform on domain-shifted asr? an extensive benchmark on air traffic control communications. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 205–212.

Towards Joint Modeling of Dialogue Response and Speech Synthesis based on Large Language Model

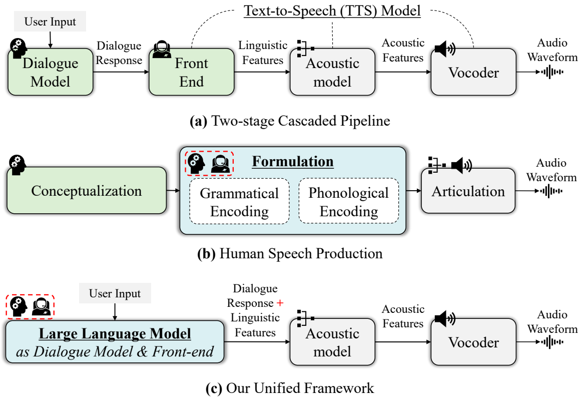
Xinyu Zhou[1] **Delong Chen**[2] **Yudong Chen**[1]
[1]Communication University of China
[2]Hong Kong University of Science and Technology
{xinyuzhou, chenyd}@cuc.edu.cn, delong.chen@connect.ust.hk

Abstract

This paper explores the potential of constructing an AI spoken dialogue system that "*thinks how to respond*" and "*thinks how to speak*" simultaneously, which more closely aligns with the human speech production process compared to the current cascade pipeline of independent chatbot and Text-to-Speech (TTS) modules. We hypothesize that Large Language Models (LLMs) with billions of parameters possess significant speech understanding capabilities and can jointly model dialogue responses and linguistic features. We conduct two sets of experiments: 1) Prosodic structure prediction, a typical front-end task in TTS, demonstrating the speech understanding ability of LLMs, and 2) Further integrating dialogue response and a wide array of linguistic features using a unified encoding format. Our results indicate that the LLM-based approach is a promising direction for building unified spoken dialogue systems.[1]

1 Introduction

As we are developing more advanced AI systems, such as Large Language Model (LLM)-based chatbots like ChatGPT and GPT-4 (OpenAI, 2023), we also hope to establish natural, seamless, and efficient communication between humans and AI systems. In addition to typing and reading through the screen, the speech channel represents a valuable alternative for interpersonal exchange, given its convenience and capacity to convey richer information than text alone. Recently, researchers from both academia and the industry have made successful attempts to concatenate AI chatbots with off-the-shelf text-to-speech (TTS) (Tan et al., 2021) modules as in Figure 1 (a), representative applications include Siri, Xiaomi Xiaoai[2] and Call Annie[3].



(a) Two-stage Cascaded Pipeline

(b) Human Speech Production

(c) Our Unified Framework

Figure 1: **A high-level comparison of different speech production processes. As noted by the red dotted boxes, the novel LLM-based unified framework proposed in this study can *"think how to respond"* and *"think how to speak"* at the same time, which aligns better with the speech production process of humans.**

However, the expressivity and interactivity of speech responses synthesized by these two-stage cascaded models are heavily limited. The reasons are two-fold. ***Firstly***, TTS modules are usually based on small language models (*e.g.,* BERT model with 0.1B parameters), which have limited capacity for understanding complex dialogue contexts. ***Secondly***, the dialogue response generation module (*i.e.,* the LLM Chatbot) and the TTS module work independently. During speech synthesizing, the TTS module can not access the information from the dialogue context, which is proven to be valuable for generating plausible and appropriate speech responses.

The current two-stage pipeline also has a fundamental difference with our understandings of the human speech production process (Levelt, 1993), where the *"grammatical encoding"* and *"phonological encoding"* are done in parallel within the "conceptualization-formulation-articulation" pro-

Xinyu Zhou and Delong Chen contributed equally.

This work is partially supported by National Social Science Fund of China (20&ZD295).

[1] Codes and datasets are publicly available at https://github.com/XinyuZhou2000/Spoken_Dialogue.

[2] https://xiaoai.mi.com

[3] https://callannie.ai

cess, as shown in Figure 1 (b). Inspired by this, we want to explore the possibility of building an AI speech dialogue system that *"thinks how to respond"* and *"thinks how to speak"* at the same time. In order to accomplish this goal, a model must possess a deep understanding of natural language and dialogue context, exhibit extensive world knowledge and commonsense, and demonstrate adequate learnability to handle text-speech joint modeling.

We hypothesize LLMs with (hundreds) billions of parameters (in comparison with BERT-based TTS front-ends (Chen et al., 2022) with only 0.3B parameters) are capable of achieving this goal. To verify this, in this paper, we provide two groups of experiments to demonstrate the possibility of building such LLM-based unified speech dialogue system.

Firstly, we get started with the prosodic structure prediction (Section 3), a typical task within the TTS text analysis front-end, to showcase the speech understanding ability of LLMs. Results show that both prompting-based ChatGPT and fine-tuning based ChatGLM (Zeng et al., 2022) model achieve competitive performance against traditional methods. We also show that LLM can utilize linguistic knowledge to improve prediction accuracy.

Secondly, we aim to further integrate a wide array of linguistic features into the model, and maintain LLM's dialogue capability at the same time (Section 4). To address the lack of a parallel dataset of dialogue response and linguistic annotations, we employ an automated dialogue context generation approach inspired by LongForm (Köksal et al., 2023), then train an LLM to produce both dialogue response speech features at the same time. Experiments show that LLM learns successfully.

2 Related Work

2.1 Human Speech Production

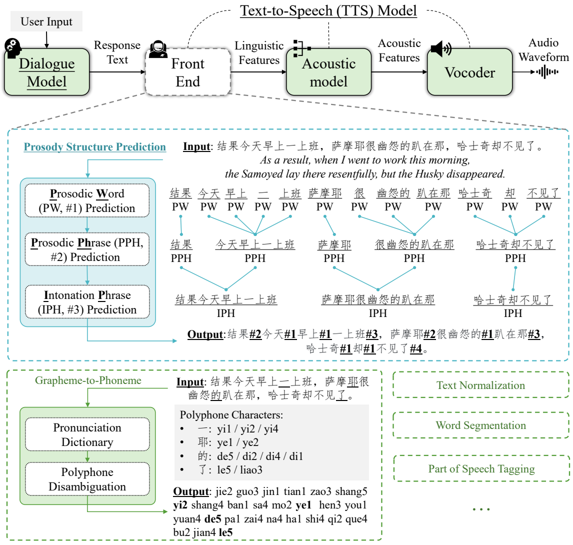
The process of human speech production is a longstanding research area. In 1993, Levelt (Levelt, 1993) proposed an encoding model for human speech production. First, concepts are generated, followed by the selection of appropriate vocabulary and the arrangement of these words according to grammatical rules. Then, the phonetics of the words are extracted in sequence, and motor programs are executed to initiate speech. The generation of spoken sentences is parallel and incremental, involving multiple stages of processing. Experiments (Schnur, 2011; Jaeger et al., 2012) prove that the phonetic planning of words begins as the grammatical structure of a sentence unfolds. Although there are many efforts to understand and explain human speech production process, TTS methods rarely take inspiration from these research results. To our best knowledge, this is the first study that attempts to build an AI system that imitates the simultaneous *"grammatical encoding"* and *"phonological encoding"* process of human speech production.

2.2 TTS Front-end and Expressivity

Typical TTS systems (Tan et al., 2021) usually consist of three main modules: front-end, acoustic model, and vocoder. The TTS Front-end models convert text into linguistic features, and are primarily BERT-based small language models, while the power of LLM is not well validated in this task yet. Hsu *et al.* (Hsu et al., 2021) and Stephenson *et al.* (Stephenson et al., 2022) have demonstrated that fine-tuning BERT can enhance the prosodic expression capabilities of TTS systems. Nevertheless, issues such as homograph ambiguity, ineffectiveness in stress, emotion and prosody still exist. Recent studies have explored the use of interactional resources (Chen, 2023), such as breathing (Székely et al., 2020), laughter (Xin et al., 2023), phonation type (Lameris et al., 2023), filled pauses and prolongations (Li et al., 2023), to improve the spontaneity and expressiveness. However, these studies have only focused on one single interactional resource, which limits their ability to capture rich and diverse subtle variations in natural conversation.

2.3 LLMs for Speech Processing

Understanding and generating speech signals are strongly related to natural language processing. With the recent explosion of LLM, many researchers in the field of speech processing also attempt to use LLMs to benefit speech or audio related tasks. AudioLM (Borsos et al., 2023) leverages a masked language model to capture the long-term structure and generate natural and coherent audio continuations given short prompts. SpeechGPT (Zhang et al., 2023), a multi-modal large language model, leverages its inherent capabilities to perceive and generate multi-modal content. PromptTTS (Guo et al., 2023) and PromptTTS2 (Leng et al., 2023) take prompts with both style and content descriptions as input to synthesize the corresponding speech.


Text-to-Speech (TTS) Model
User Input
Dialogue Model
Response Text
Front End
Linguistic Features
Acoustic model
Acoustic Features
Vocoder
Audio Waveform
Prosody Structure Prediction
Prosodic Word (PW, #1) Prediction
Prosodic Phrase (PPH, #2) Prediction
Intonation Phrase (IPH, #3) Prediction
结果 今天 早上 一 上班 萨摩耶 很 幽怨的 趴在那 哈士奇 却 不见了
PW PW PW PW PW PW PW PW PW PW PW PW
结果 今天早上一上班 萨摩耶 很幽怨的趴在那 哈士奇却不见了
PPH PPH PPH PPH PPH
结果今天早上一上班 萨摩耶很幽怨的趴在那 哈士奇却不见了
IPH IPH IPH
Grapheme-to-Phoneme
Pronunciation Dictionary
Polyphone Disambiguation
Text Normalization
Word Segmentation
Part of Speech Tagging
. . .

Input: 结果今天早上一上班，萨摩耶很幽怨的趴在那，哈士奇却不见了。
As a result, when I went to work this morning, the Samoyed lay there resentfully, but the Husky disappeared.

Output:结果**#2**今天**#1**早上**#1**一上班**#3**，萨摩耶**#2**很幽怨的**#1**趴在那**#3**，哈士奇**#1**却**#1**不见了**#4**。

Input: 结果今天早上一上班，萨摩耶很幽怨的趴在那，哈士奇却不见了。

Polyphone Characters:
- 一: yi1 / yi2 / yi4
- 耶: ye1 / ye2
- 的: de5 / di2 / di4 / di1
- 了: le5 / liao3

Output: jie2 guo3 jin1 tian1 zao3 shang5 **yi2** shang4 ban1 sa4 mo2 **ye1** hen3 you1 yuan4 **de5** pa1 zai4 na4 ha1 shi4 qi2 que4 bu2 jian4 **le5**

Figure 2: **Standard pipeline of current spoken dialog systems.** A dialogue model generates a response to user input, and the TTS model (front-end→acoustic model→vocoder) converts text to audio subsequently.

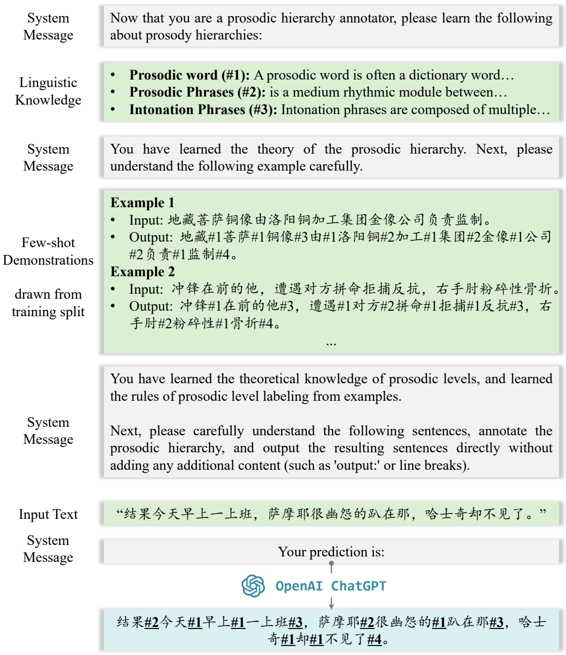
3 Prosodic Structure Prediction based on Large Language Model

Prosodic Structure Prediction (PSP) is a typical task in the Chinese TTS front-end (Chen et al., 2022), among others like grapheme-to-phoneme prediction, text-normalization, word segmentation, part-of-speech tagging, etc. As illustrated in Figure 2, a PSP model needs to identify multiple levels of prosody hierarchy, including Prosodic Word (PW), Prosodic Phrase (PPH), and Intonation Phrase (IPH), which can be denoted as #1, #2, and #3 respectively in the output sentence.

Prosodic structure is one of the most important linguistic features in Chinese TTS, and it is strongly related to the syntax of the sentence. In this section, we want to validate whether the LLMs, which have been well-proved to have superior semantic understanding abilities, but are trained on the text-only corpus, can handle this speech-related task. In the following, we present two methods for adapting LLM to the PSP task: prompting (Section 3.1), and fine-tuning (Section 3.2).

3.1 Prompting LLM for Prosodic Structure Prediction

Prompting is the most convenient way to adapt an instruction-following LLM to new tasks. In Figure 3, we present an overview of our proposed prompt structure for PSP on LLM, which consists of linguistic knowledge of Chinese prosodic structure, few-shot demonstrations as in-context learning examples, input sentence, and interleaved system messages to explain each part to the LLM.



Now that you are a prosodic hierarchy annotator, please learn the following about prosody hierarchies:

- **Prosodic word (#1):** A prosodic word is often a dictionary word…
- **Prosodic Phrases (#2):** is a medium rhythmic module between…
- **Intonation Phrases (#3):** Intonation phrases are composed of multiple…

You have learned the theory of the prosodic hierarchy. Next, please understand the following example carefully.

Example 1
- Input: 地藏菩萨铜像由洛阳铜加工集团金像公司负责监制。
- Output: 地藏#1菩萨#1铜像#3由#1洛阳铜#2加工#1集团#2金像#1公司#2负责#1监制#4。

Example 2
- Input: 冲锋在前的他，遭遇对方拼命拒捕反抗，右手肘粉碎性骨折。
- Output: 冲锋#1在前的他#3，遭遇#1对方#2拼命#1拒捕#1反抗#3，右手肘#2粉碎性#1骨折#4。

...

You have learned the theoretical knowledge of prosodic levels, and learned the rules of prosodic level labeling from examples.

Next, please carefully understand the following sentences, annotate the prosodic hierarchy, and output the resulting sentences directly without adding any additional content (such as 'output:' or line breaks).

“结果今天早上一上班，萨摩耶很幽怨的趴在那，哈士奇却不见了。”

Your prediction is:

结果**#2**今天**#1**早上**#1**一上班**#3**，萨摩耶**#2**很幽怨的**#1**趴在那**#3**，哈士奇**#1**却**#1**不见了**#4**。

Figure 3: **Our proposed prompt structure for LLM (ChatGPT)-based prosodic structure prediction.** We incorporate expert linguistic knowledge and few-shot demonstrations to enable LLMs to perform the prosodic structure prediction task.

Linguistic Knowledge contains formal definitions of Chinese prosodic structure summarized from recognized research literature (Cao, 2003). It describes distinct characteristics and positions within sentences and phrases of three levels of prosodic structure in Chinese.

Few-shot Demonstration provides input-output pairs to LLM for in-context learning. Examples are either randomly drawn from the training split or carefully selected based on the assessment of their representativity and quality from the linguistic perspective. The maximum number of few-shot demonstrations is 16, as more examples would exceed the context window length of LLM.

3.2 Fine-tuning LLM for Prosodic Structure Prediction

Context window length is a crucial limit for prompting-based methods, as it prohibits the LLM from learning from more (than 16) training examples. Furthermore, all model parameters remain fixed and unlearnable, resulting in limited learning capacity. To address these constraints, we propose the fine-tuning of a Large Language Model (LLM) to enhance Prosodic Structure Prediction learning from a substantially larger number of training examples, up to 8,000.

It has been proved that using a Pretrained Language Model (PLM) such as BERT (Devlin et al., 2018) to be the initialization of the PSP model is beneficial, such as SpanPSP (Chen et al., 2022), J-TranPSP (Shen et al., 2022), and MLC-PSP (Chen et al., 2023), our methodology of fine-tuning LLM has some difference from them. Despite the difference in model scale (0.1B vs 6B), previous BERT-based methods regard PSP as a *token classification* problem, where the model needs to determine whether there is a prosodic boundary after each character and what level is it. In contrast, here we formalize PSP as a sequence-to-sequence (Seq2seq) prediction task, where input x is the raw sentence and the output y is a string of character sequence with "#n" ($n \in \{1, 2, 3\}$) notation of prosodic structure.

We apply standard cross-entropy loss for auto-regressive language modeling as the learning objective, and we only calculate the loss on output tokens. We add a prefix c of "Please perform prosodic prediction on the given sentence:" into the input for better initialization. The following is the loss function $\mathcal{L}(\theta)$ of the LLM θ, where N is the number of training samples: $\mathcal{L}(\theta) = -\sum_{i=1}^{N} \log p_\theta(y_i|x_i, c_i)$.

3.3 Experiment Setup

Dataset. We utilize the DataBaker open-source Chinese Standard Mandarin Speech Corpus[4], which contains 10-hour speech recordings of 10,000 sentences with an average length of around 16 words per sentence. It was articulated by a single Chinese female speaker. The corpus also encompasses diverse domains, including news, novels, technology, entertainment, etc.

Furthermore, the dataset is enriched with various linguistic annotations, including character, pinyin, and prosodic hierarchy information, as well as phoneme level interval and boundary data. Annotations for prosodic hierarchy comprise PW (#1), PPH(#2), IPH (#3), and the end of a sentence (#4). We discard the #4 annotations as every sample is a single sentence and only has a "#4" in the end. The remaining labels collectively form a hierarchical prosodic tree structure with three distinct layers.

We split 10k samples with an 8:1:1 ratio for training, validation, and testing. Few-shot demonstrations are drawn from the 8k training split. For the "random" selection setting, we sample demonstrations randomly three times and report the averaged performance.

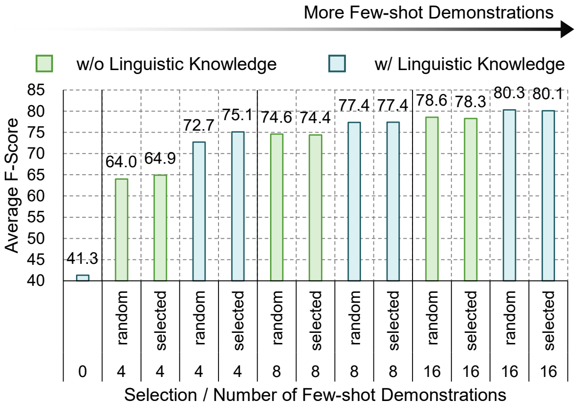


Figure 4: **Ablation study of prompting ChatGPT based PSP**. We compared different numbers of few-shot demonstrations, selection of few-shot demonstrations, and variants of with (w/) or without (w/o) linguistic knowledge.

Implementation Details. For the prompting-based method, we test the OpenAI's `text-davinci-002` API (ChatGPT) and the ChatGLM2-6B model. For the fine-tuning-based method, we only unitize the ChatGLM2-6B model due to the limitation of computational resources. We apply P-tuning-v2 (Liu et al., 2022) for parameter-efficient fine-tuning using the official codebase [5]. We used a single NVIDIA A100 GPU for both training and testing.

3.4 Ablation Study

We first provide ablations for the prompting-based approach. Following previous works on PSP tasks, we use F-Score as the evaluation metric. As it can be seen from Figure 4, the number of few-shot demonstrations makes a significant impact. Four in-context examples lead to +22.7% improvements to zero-shot setting (41.3%→64.0%), while incorporating linguistic knowledge brings another around +8.7% improvements (→72.7%), and further, when swapping random demonstrations to carefully selected high-quality demonstrations, we receive another +2.4% performance gain (→75.1%).

We further ablate different levels of linguistic knowledge in Table 2. It shows that linguistic expert knowledge plays a crucial role in the prediction of Prosodic Phrase (#2) and Intonational Phrase (#3). We hypothesize it is caused by different difficulties of #1 to #3 predictions – #1 usually appears at word boundaries, while identifying #2 and #3 is

[4] `https://www.data-baker.com/data/index/TNtts`

[5] `https://github.com/THUDM/ChatGLM2-6B/tree/main/ptuning`

Table 1: **Benchmarking of PSP Models**. We compared the F-Score of the traditional BERT-based method SpanPSP and our newly proposed LLM-based methods (prompting or fine-tuning) using two LLMs with different scales (ChatGPT and ChatGLM).

Model (#Parameters)	Variation	PW #1	PPH #2	IPH #3	Average
SpanPSP (0.1B)	Databaker Pretrained	**96.35**	69.34	65.64	77.11
	PeopleDaily Pretrained	89.20	71.08	79.12	79.80
ChatGPT (175B)	Knowledge Only	61.87	27.27	34.78	41.31
	16 Random Examples	88.51	69.40	77.91	78.61
	Knowledge + 16 Selected Examples	90.12	69.40	**80.85**	80.12
ChatGLM2-6B	Knowledge + 16 Selected Examples	N/A	N/A	N/A	N/A
	Fine-tuned	93.86	**73.28**	80.00	**82.38**

Table 2: **Ablations of removing each level of linguistic knowledge.** Expert knowledge is especially useful for higher levels of prosodic structure prediction (*i.e.,* PPH and IPH).

Knowledge Ablation	PW #1 F-Score	PPH #2 F-Score	IPH #3 F-Score	Average F-Score
w/o #1	**88.54**	64.66	78.30	77.17
w/o #2	87.57	**61.63**	79.09	76.10
w/o #3	87.72	64.69	**78.14**	76.85
Default (all)	**88.14**	**65.03**	**79.52**	**77.56**

not that straightforward.

3.5 Benchmarking LLM-based PSP

Baseline. SpanPSP (Chen et al., 2022) is a classical character-level BERT-based model for the PSP task, which is based on a relatively small language model `bert-base-chinese`[6] with only 0.1B parameters. We use their official checkpoints and codebase[7] for evaluation.

We provide benchmarking results in Table 1. It reveals that carefully crafted linguistic knowledge and selected examples (*i.e.,* "Knowledge + 16 Selected Examples" variation) enable ChatGPT to outperform the traditional method SpanPSP (80.12% vs. 79.80%), but such a prompting-based learning strategy failed (N/A) at smaller open-source LLM (ChatGLM) due to its limited instruction-following ability. However, it shows that fine-tuning smaller LLM can outperform prompting larger LLM (82.38% vs. 80.12%), as it can access more training samples (8k training set vs. the maximum of 16 in-context examples).

4 Joint Prediction of Dialogue Response and Linguistic Features

In the last section, we have shown some positive results proving LLMs are competitive at a typical front-end task in Chinese TTS. Here in this section, we want to go beyond just a single task in TTS, and validate the possibility of building a LLM-based system that can handle versatile tasks in Chinese TTS front-end (Figure 2), and also maintain its dialogue capability of generating coherent responses to user queries simultaneously. By implementing such a model that jointly predicts dialogue response (*i.e., "think how to respond"*) and linguistic features (*i.e., "think how to speak"*) at the same time, we could have an AI system that communicates with humans in a way that is more similar to the human speech production process (Levelt, 1993), where the "grammatical encoding" and "phonological encoding" are done in parallel.

4.1 Methodology

Dialogue Context Generation. Our objective is to investigate the feasibility of constructing a unified model capable of simultaneously generating coherent responses to user queries in dialogues and diverse fine-grained linguistic features for TTS. Unfortunately, the DataBaker dataset only comprises isolated sentence recordings, and there are not any other datasets having dialogue context and parallel speech recordings or annotations. Drawing inspiration from the LongForm approach (Köksal et al., 2023), we prompt ChatGPT to anticipate the dialogue context and transform it into a dataset of single-turn dialogues:

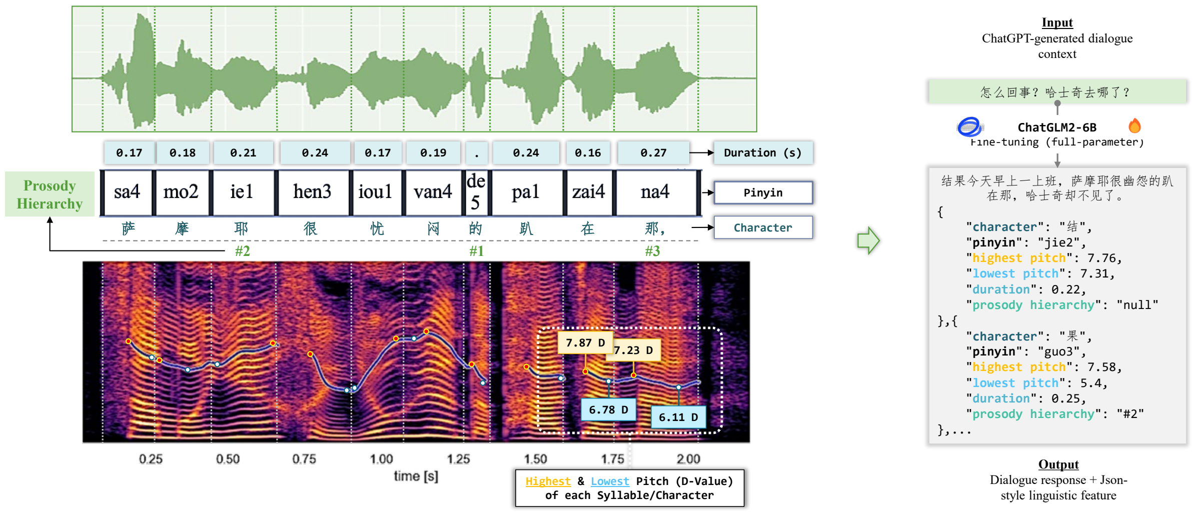
Prompt for Dialogue Context Generation

```
### System Message:
Please generate the most likely sentence spoken by A based on B's response.
### User:
A:
B: "When I went to work this morning, the Samoyed lay there resentfully, but the Husky disappeared."
---
### ChatGPT:
A: "What's going on? Where did the Husky go?"
```

Linguistic Feature Extraction. As shown in Figure 5 left, we automatically extract the follow-

[6] https://huggingface.co/bert-base-chinese

[7] https://github.com/thuhcsi/SpanPSP



```
结果今天早上一上班，萨摩耶很幽怨的趴
在那，哈士奇却不见了。
{
    "character": "结",
    "pinyin": "jie2",
    "highest pitch": 7.76,
    "lowest pitch": 7.31,
    "duration": 0.22,
    "prosody hierarchy": "null"
},{
    "character": "果",
    "pinyin": "guo3",
    "highest pitch": 7.58,
    "lowest pitch": 5.4,
    "duration": 0.25,
    "prosody hierarchy": "#2"
},...
```

Figure 5: **Left:** an overview of linguistic feature extraction. We automatically extract a wide array of linguistic features, including character, duration, pinyin, prosody hierarchy, highest pitch and lowest pitch ($\mathcal{D}$-Value). **Right:** the illustration of data formatting. We encode extracted linguistic features into JSON-formatted strings, such that they can be fed to LLM directly as learning targets.

ing four categories of linguistic attributes: characters, their corresponding duration, pinyin (phonetic transcription representing character pronunciation), prosodic hierarchy, and the highest and lowest pitch values ($\mathcal{D}$-Value). The use of $\mathcal{D}$-value is inspired by Shen Jiong's theory (Shen, 1985): the D-value is a logarithmic scale used to describe pitch and quantifies the relationship between a pitch (F) in Hertz (Hz) and a reference frequency (F_0). It provides a measure of pitch variation, which is especially useful for observing pitch contours in speech. The formal definition of ($\mathcal{D}$-Value) is: $\mathcal{D} = 5 \times \log_2(F/F_0)$.

Data Formatting. As shown in the right side of Figure 5, we format extracted linguistic features into a string of JSON-style dictionaries, and concatenate it with the response text generated by ChatGPT, together serving as the learning target. Such implementation realizes joint learning in a seamless way and enjoys simplicity over the traditional method, where different types of outputs (response, various linguistic features) are usually produced by different models, or one model with different task-specific heads (Bai and Hu, 2021). Our approach also shares some similarities with recent advances in LLM research, such as RT-2 (Brohan et al., 2023) from Google DeepMind, where the LLM are trained to produce not only natural language output but also some continuous values.

4.2 Experiment Setup

Training. Empirically, we found that P-tuning (as used in fine-tuning-based PSP in Section 3.2) failed to learn how to generate dialogue response and JSON-style linguistic features. Therefore, for this section, we turn to use full-parameter fine-tuning to enable more learning capacity. We use 4-bit quantization to boost memory efficiency, as JSON-style encoding takes much longer context than that in the PSP task (maximum 1.6k tokens vs. 128 tokens).

Table 3: **Evaluation result of LLM produced linguistic features.** Most of the output JSON-style is incorrect grammar and parsable, and the majority of these parsable characters can be matched with ground truth. However, we can observe a notable train-test performance gap, meaning that the model suffers from overfitting.

	Parsable Samples	Matched Characters	Matched Pinyin	Matched Prosody
Training Split	95.90%	86.88%	98.79%	97.75%
Testing Split	89.70%	69.26%	86.29%	77.70%

Testing. Based on our data formatting (Figure 5), given a user utterance as input, the model will first give its dialogue response, then the JSON-style linguistic feature of each word in the response sentence subsequently. However, this poses a challenge for the evaluation of the linguistic feature, since for unseen testing quires, the LLM-outputted response would be different from the ground truth response, thus making them not comparable. To solve this issue, we use the ground truth response as a generation prefix, and then try to parse the generated dictionary and compare them with ground truth linguistic features.

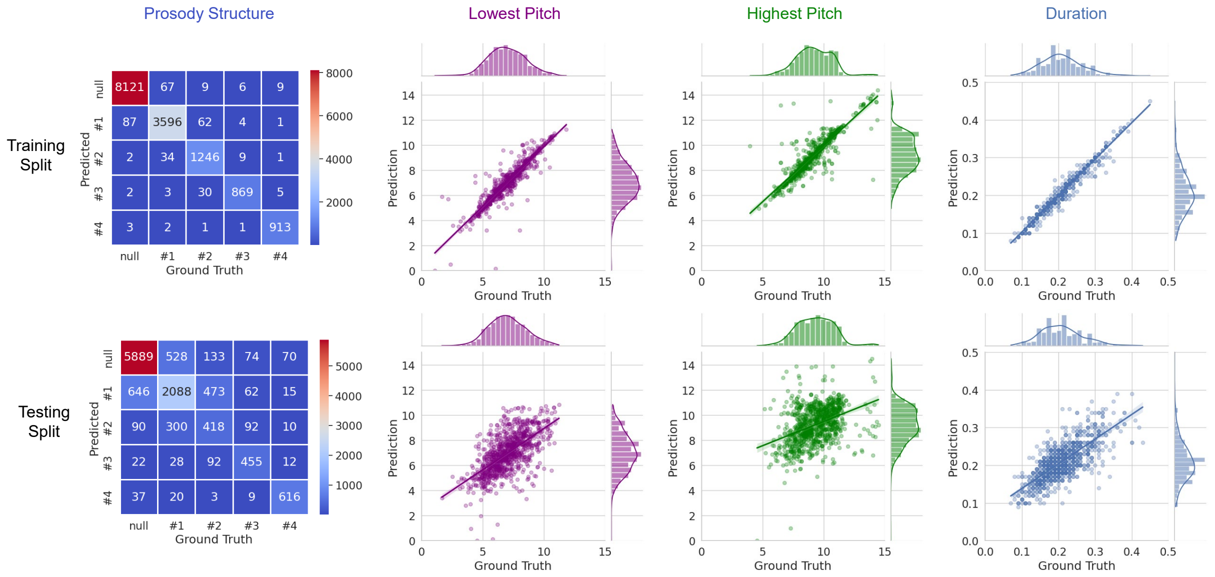


Figure 6: **Evaluations results of fine-tuning ChatGLM2-6B on joint dialogue response and linguistic features.** Visualizations show that the model fits the training data quite well, showing the feasibility of our proposed joint learning approach. But possibly due to insufficient dataset scale, the generalization ability of the model is somewhat weak.

4.3 Experiment Results

In Table 3, we provide the evaluation result of 1k testing samples and randomly sampled 1k training samples. As can be seen, the model performed quite well on the training set, achieving 95.90% parsable samples, 86.88% matched characters, and 98.79% matched Pinyin, showing that it successfully fit the JSON-format data. When tested on unseen samples, the model successfully generated Json-style linguistic features with an 89.70% success rate. However, due to the limited capacity of the small LLM, missing characters were frequently observed, resulting in only 69.26% of characters from the ground truth being found in the generated results. Within those matched characters, the model achieved an 86.29% success rate in producing matched Pinyin, and a 77.70% success rate of matched prosody structure annotation.

In Figure 6, we visualize the model predictions versus the ground truth of continuous values. Again, we observed that the model effectively fit the training set and demonstrated a certain level of generalization ability when applied to new data.

5 Discussion

In this study, we presented two groups of experiments to validate the possibility of building an LLM-based spoken dialog system that *"thinks how to respond"* and *"thinks how to speak"* at the same time. In the first group of experiments, we proved that LLM is a competitive prosodic structure predictor, which means that its rich world knowledge and semantic understanding ability acquired from text-only pretraining can transferred to benefit speech-related tasks. Based on this observation, we further involve many other linguistic features in our second group of experiments and further proved that it is possible for LLM to learn to generate dialogue response and speech features at the same time. However, there are still several noticeable limitations of this study, which are summarized from the following four perspectives:

Model Perspective. The training cost of LLM is high. Additionally, the auto-regressive decoding of Json-style linguistic features is quite time-consuming – processing a single sentence and its linguistic features takes at least 15 seconds (for long sentences, it could be 40+ seconds).

Data Perspective. The current training dataset consists of only 8k samples, which is insufficient and has led to a substantial over-fitting phenomenon. Speech style in the dataset is limited, as it was sourced from a single speaker, primarily containing formal read recordings, lacking the nuances inherent in natural conversations.

Expressivity Perspective. According to interactional linguistics studies (Couper-Kuhlen and Selting, 2017), finer-grained annotation system by comprehensively and meticulously annotating the collected speech dataset with interactional resources like voice quality, phonation type, breath patterns, repair, interjection, pause, prolongation, etc, will further increase the expressivity.

System Perspective. It is important to note that so far this study does not include subsequent acoustic models and vocoders and is not able to generate audio waveform, we only use speech linguistic features to represent speech information.

References

Zilong Bai and Beibei Hu. 2021. A universal bert-based front-end model for mandarin text-to-speech synthesis. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6074–6078. IEEE.

Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. 2023. Audiolm: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. 2023. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*.

Jianfen Cao. 2003. Prediction of prosodic organization based on grammatical infomation (in chinese). *Journal of Chinese Information Processing*, (41-46).

Jie Chen, Changhe Song, Deyi Tuo, Xixin Wu, Shiyin Kang, Zhiyong Wu, and Helen Meng. 2023. Improving mandarin prosodic structure prediction with multi-level contextual information. *arXiv preprint arXiv:2308.16577*.

Xueyuan Chen, Changhe Song, Yixuan Zhou, Zhiyong Wu, Changbin Chen, Zhongqin Wu, and Helen Meng. 2022. A character-level span-based model for mandarin prosodic structure prediction. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7602–7606. IEEE.

Yudong Chen. 2023. Studies of prosodic expressions in interactive language: A review (in chinese). *Contemporary Linguistics*, 25(197-222).

Elizabeth Couper-Kuhlen and Margret Selting. 2017. *Interactional linguistics: Studying language in social interaction*. Cambridge University Press.

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*.

Zhifang Guo, Yichong Leng, Yihan Wu, Sheng Zhao, and Xu Tan. 2023. Prompttts: Controllable text-to-speech with text descriptions. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.

T Florian Jaeger, Katrina Furth, and Caitlin Hilliard. 2012. Incremental phonological encoding during unscripted sentence production. *Frontiers in Psychology*, 3:481.

Abdullatif Köksal, Timo Schick, Anna Korhonen, and Hinrich Schütze. 2023. Longform: Optimizing instruction tuning for long text generation with corpus extraction. *arXiv preprint arXiv:2304.08460*.

Harm Lameris, Shivam Mehta, Gustav Eje Henter, Joakim Gustafson, and Éva Székely. 2023. Prosody-controllable spontaneous tts with neural hmms. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Yichong Leng, Zhifang Guo, Kai Shen, Xu Tan, Zeqian Ju, Yanqing Liu, Yufei Liu, Dongchao Yang, Leying Zhang, Kaitao Song, et al. 2023. Prompttts 2: Describing and generating voices with text prompt. *arXiv preprint arXiv:2309.02285*.

Willem JM Levelt. 1993. *Speaking: From intention to articulation*. MIT press.

Weiqin Li, Shun Lei, Qiaochu Huang, Yixuan Zhou, Zhiyong Wu, Shiyin Kang, and Helen Meng. 2023. Towards spontaneous style modeling with semi-supervised pre-training for conversational text-to-speech synthesis. *arXiv preprint arXiv:2308.16593*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Tatiana T Schnur. 2011. Phonological planning during sentence production: Beyond the verb. *Frontiers in Psychology*, 2:319.

Binbin Shen, Jian Luan, Shengyan Zhang, Quanbo Shen, and Yujun Wang. 2022. J-tranpsp: A joint transition-based model for prosodic structure prediction, word segmentation and pos tagging. In *2022 13th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 280–284. IEEE.

Jiong Shen. 1985. Pitch range of tone and intonation in beijing dialect (in chinese). *Experiments on Beijing Dialect. Beijing: Peking University*, pages 73–130.

Brooke Stephenson, Laurent Besacier, Laurent Girin, and Thomas Hueber. 2022. Bert, can he predict contrastive focus? predicting and controlling prominence in neural tts using a language model. *arXiv preprint arXiv:2207.01718*.

Éva Székely, Gustav Eje Henter, Jonas Beskow, and Joakim Gustafson. 2020. Breathing and speech planning in spontaneous speech synthesis. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7649–7653. IEEE.

Xu Tan, Tao Qin, Frank Soong, and Tie-Yan Liu. 2021. A survey on neural speech synthesis. *arXiv preprint arXiv:2106.15561*.

Detai Xin, Shinnosuke Takamichi, Ai Morimatsu, and Hiroshi Saruwatari. 2023. Laughter synthesis using pseudo phonetic tokens with a large-scale in-the-wild laughter corpus. *arXiv preprint arXiv:2305.12442*.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*.

Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. 2023. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. *arXiv preprint arXiv:2305.11000*.

Using Whisper LLM for Automatic Phonetic Diagnosis of L2 Speech: A Case Study with French Learners of English

Nicolas Ballier[1 2], Adrien Méli [2], Maelle Amand [3], Jean-Baptiste Yunès [4]
[1]LLF / [2]CLILLAC-ARP / [4]IRIF Université Paris Cité, F-75013 Paris, France
[3] CeRES / Université de Limoges 39E rue Camille Guérin F-87000 Limoges
{nicolas.ballier, jean-baptiste.yunes}@u-paris.fr, adrienmeli@gmail.com
maelle.amand@unilim.fr

Abstract

This paper reports on a pilot study to use Whisper's large language model (LLM) as a tool for potential representation of segmental (phone) pronunciation errors. We compared the performance of the transcription outputs for the various models developed by the automatic speech recognition (ASR) system Whisper (Radford et al., 2022) ranging from 39 to 1,550 million parameters. We investigated 38 recordings of two paragraphs from Conrad's *Typhoon*. The whisper transcriptions were compared to the original text that was read by these second-year French undergraduates. We used WER (Word Error Rate) and Levenshtein distance to assess the various graphic representations of Conrad's reference text. We show how the differences can be transformed into operationalised feedback for learners. We used expert phonetic knowledge to check the plausibility of the phonetic interpretation with the signal (in particular the recall of H dropping produced by French learners). Our findings suggest that the transcriptions produced by the `medium` model converge with what a native speaker understands and that the `tiny` model produces alternate transcriptions that are plausible candidates for learner errors.

1 Introduction

Whisper is an audio multilingual large language model (mLLM) which can be used for two types of tasks, transcription (speech to text) and translation (only to English). Using thousands of hours of training data, mostly from Librispeech (Panayotov et al., 2015), a dataset of read speech of public domain books, Whisper has been trained with both multilingual data and English only data. Several models have been created with an increasing number of parameters, as listed in Table 1 (Radford et al., 2022). Probably because of Named Entity Recognition (NER) issues as acknowledged in (Radford et al., 2022), proper nouns (but other tokens as well) can undergo what we call a *retranscription*, i.e., that differs from the original text but that is phonetically consistent with the speech input, e.g., *Macquaire* instead of the expected *McWhirr*.

In this paper, we follow the standard phonological convention that indicates graphemes (letters) with angled brackets(<>), realisations in square brackets and phonemes (or targets) with slanted bars (//). Our research questions are as follows: do ASR retranscriptions differ from one Whisper model to the next, and how realistic are they as (re)interpretation of learner phonetic realisations?

Previous research has suggested that the Whisper retranscriptions vary across Whisper models (**?**) while trying to be faithful to the phonetic input of a foreign pronunciation. This paper essentially assesses two Whisper models (`tiny` and `medium`) in their ability to capture relevant L2 pronunciation errors in classroom or computer-assisted learning environments. We want to test the hypothesis that the `tiny` model is more likely to retranscribe pronunciation errors than the `medium` model. Our hypothesis is somehow counter-intuitive as the lowest model with the least number of parameters is chosen to be the most efficient to represent / to emulate learner representation or the learner data as perceived by native speakers. We are working on the discrepancy between the transcriptions integrated condition with the reference target hypothesis.

We first provide a quantitative analysis of these discrepancies before analysing the fine phonetic renditions of the different files. Two professionals trained in phonetics analysed the phonetic data and tried to extract one of the most striking features from a phonetic point of view in order to be used as feedback for learners : H-dropping, namely, the lack of aspiration. Since /h/ is not part of the phonemic inventory of French, most learners either omit the sound or substitute it with a glottalisation (Exare, 2022). The two operations were carried out independently. We then analyse the extent in which

Whisper's graphic renditions match the phonetic interpretation of the learners' mispronunciations.

The rest of the paper is structured as follows: Section 2 presents the previous research carried out on automatic speech recognition with learner data. Section 3 presents the data we tested and the metrics we used. Section 4 presents our results and Section 5 discusses them.

2 Previous Research

The use of ASR in pronunciation training dates back to the 1990s. A preliminary study pioneered the use of ASR in L2 pronunciation (Rogers et al., 1994), showing that ASR helped improve intelligibility in the learner's L2 and that the improved targeted phonetic contrasts (/iː/ vs. /ɪ/, /θ/ vs. /s/) were also found in untrained words. Watson et al. (1989) compared human and ASR evaluations of speech quality. Some explored ways to integrate ASR in pronunciation training programs (Dalby and Kewley-Port, 1999), while others focused on the creation of feedback derived from the ASR transcriptions. More recent studies (Inceoglu and Lim, 2023) used Google's ASR to measure the intelligibility of L2 speech (Taiwanese L1, English L2) and concluded that the rating-agreement between the ASR and native speakers mostly depended on both the individual speakers and the speech style (i.e., word lists, read text or more natural speech). Similar systems have been developed with Open Source release, such as KALDI (Povey et al., 2011), Vosk [1], wav2vec 2.0 (Baevski et al., 2020), and others for ASR models.

ASR models have also been applied to the analysis of L2 speech. Previous studies focused on the discrepancies between the ASR output of L2 speech and the expected target (Chanethom and Henderson, 2022; Inceoglu et al., 2020). In this respect, an important contribution is an analysis based on Weinberger's Speech Accent Archive (Weinberger, 2015), which considers native and non-native varieties of English alike, to analyse how the ASR system *Otter.ai* performs in investigating the effect of syllable structures on the realisations of clusters and of vowel substitutions in relation to vowel spaces (**?**).

To the best of our knowledge, our paper is the first paper that uses Whisper to investigate learner speech and, more generally, that compares the performance of several models within the same ASR system.

Size	Parameters
tiny	39 M
base	74 M
small	244 M
medium	769 M
large	1550 M
large-v2	1550 M

Table 1: Whisper's main models for speech recognition, after (Radford et al., 2022)

3 Materials and Method

3.1 Whisper Parameters and Outputs

Whisper uses the Encoder-Decoder Transformer architecture and takes audio as input, chunked in 30s windows and converted to a log-Mel spectrogram. Whisper is trained to predict the corresponding text (Radford et al., 2022) and its translation into English. Transcription and translation are the two main tasks, but Whisper can also provide language identification. We tested the learner speech with Whisper's different models. Table 1 lists the corresponding parameters of these models. After a transcription, each Whisper model outputs files in the Hugging Face implementation with with or without time stamps. A .json file includes the metadata of the prediction outputs for each segment (the average log probability, the compression ratio and the probability of the absence of speech).

3.2 Selected Reference Target for Learner Data

38 graduate-level learners of English at a French University were asked to read the first two paragraphs of chapter 2 from Joseph Conrad's *Typhoon* (1902).[2]. The text counted 408 words with 17 sentences. It was deemed suitable for L2 speakers with a C1 level by CEFR standards by the CATHOVEN text analyser [3] due to the richness of the vocabulary and complexity of the sentences. The high cognitive load required to read the text was expected to highlight pronunciation difficulties that are not fully mastered by the L2 learners (Christodoulides, 2016). These two paragraphs contain a wide array of potential pronunciation difficulties for French

[1] https://alphacephei.com/vosk/

[2] The students were warned that the term *Chinaman* was considered offensive and that it should not be used today when referring to a person.

[3] https://hub.cathoven.com/?scene=analyser

L2 learners (voicing of intervocalic <s> in *precisely*, H-dropping of initial and medial /h/ or H-intrusion (*hair* for *air)*, unstable vowel length contrast (*(h)it* instead of *heat*), lack of initial aspiration for voiceless plosives (*pigtail* is understood as *big tail*, vowel reduction, misplacement of lexical stress...).

3.3 Metrics

To analyse the retranscriptions produced by Whisper, we used word error rate (WER) a standard metric for Automatic Speech Recognition systems and Levenshtein distance (Levenshtein et al., 1966), as produced by the R package {phonics}(Howard II, 2020), since it offers insights into the discrepancy between the target hypothesis and the learner realisation, and a graphic rendition of the learner realisation produced by the different models.

4 Results

4.1 Selecting the Optimal Model for Leaner Data Transcription

In this section, we report our findings on the Whisper .txt outputs by ASR model. Figure 1 displays the boxplots corresponding to the WER of the different Whisper models. No significant difference in performance (WER) was found between the models specifically trained with English data (whether `tiny.en` or `medium.en`) and the multilingual models. A t-test revealed no significant difference between the multilingual `tiny` model and English-only `tiny.en` model (t = 2.1947, df = 37, p-value = 0.03454). While the WER between the multilingual `tiny` model and the `medium` model was deemed significant (t-test : t = 7.3121, df = 37, p-value < 0.001), that between the `medium` and the `medium.en` was not significant. Nevertheless, a more detailed comparison revealed that the `tiny` model produces a higher WER than the `tiny.en` model, wheareas the `medium.en` model had higher error rates than the `medium` model. This seems to suggest that the `tiny` model is the most efficient model in capturing non-native pronunciation oddities, while the equivalent model based on English only seems to normalise such oddities.

4.2 Number of Retranscriptions and Model Size

String distance was also examined between the models, and more specifically, the number of additions and the number of tokens that were outputted by each model but were not in the reference text. We found that the number of added tokens decreased almost linearly with the log of the number of parameters for each model from `tiny` to `medium` (Figure 2).

The different models produce different types of respelling (and in varying quantities). This is true for the `tiny` vs. `tiny.en` models but also for the `medium` vs. `tiny` models. We tried to test the separability of the tokens that were retranscribed by these models and used a Venn's diagram to categorise the different model reinterpretations of the same acoustic signal (Figure 4). The retranscriptions of the different models are not mutually exclusive, as the `medium` and the `tiny` models share 16.1 % of their retranscriptions, but they must not be understood as a simple numerical decrease of alternative respellings across models. In fact, they include different tokens that are not in the reference text. Further research is needed to investigate why the different Whisper models produce different graphemic representations, since the models are based on the same (sub)token dictionary after the Byte-pair encoding.

4.3 Plausibility of the Whisper Respelling

This subsection tentatively reports on the precision of the retranscription, by detailing the phonetic interpretation of respellings. The 38 `tiny` models produced 832 tokens differing from the original reference text, including one recording transcribed exclusively into French. Some hapaxes corresponded to mispronunciations such as <alphnicate> for *half-naked*,which are consistent with common features amongst non-native speakers: h-dropping (Exare, 2017), monophthonguisation of <a> in <naked> with harmonisation with the second vowel ([nikit]instead of /neɪkɪd/) the devoicing of final consonants (here, /t/ for /d/, cf. (Hutin et al., 2020)).

4.4 Precision and Recall

Assessing precision and recall of the phonetic error detection means answering the following questions : how many of the Whisper retranscriptions point to an actual pronunciation error (precision) and how many of the learners' pronunciation errors were captured in the Whisper transcriptions (recall)? In this paper, we do not address the precision and recall of the phonetic errors by the system, as it would require intensive manual phonetic annota-

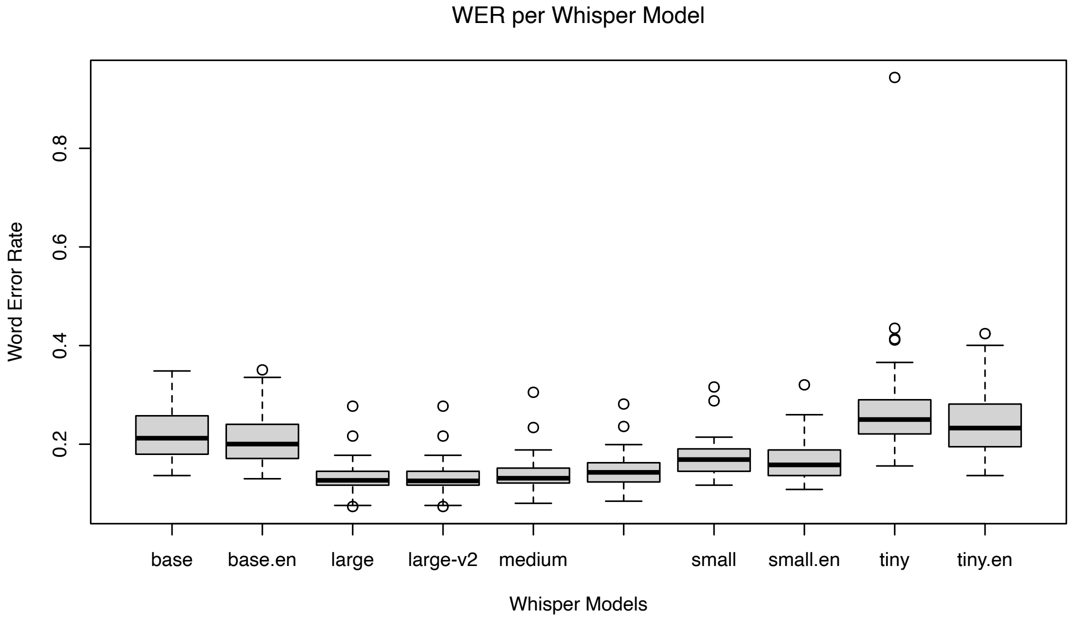


Figure 1: Dispersion of the WER across speakers for each Whisper model, trained on English data only (.en) or on multilingual data

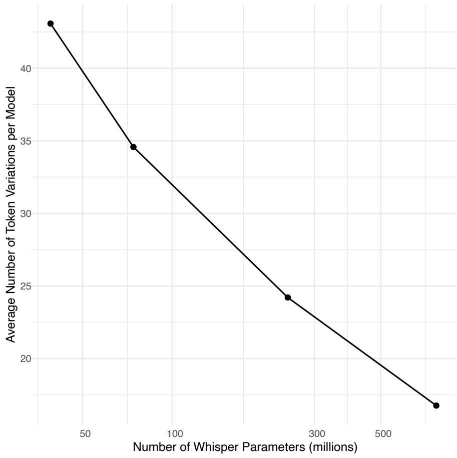


Figure 2: Relation between the numbers of parameters (log scale) of the `tiny`, `base`, `small` and `medium` models and the detection of Pronunciation Errors Candidates signalled by spelling variants or token additions

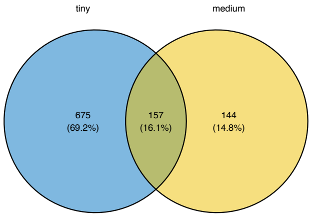


Figure 3: Venn's diagram of the common retranscriptions between the `medium` and the `tiny` models

tion of all the files, but we offer a critical diagnosis of some of the most frequent retranscriptions we observed, especially discussing those which potentially emulated some misunderstanding with a native. For precision, we analysed the 13 frequent added tokens and noticed false alarms for less frequent items as well. The presence of a reduced vowel for the realisation of *seaman* led to the transcription of the token as *semen* and as *seamen* (it should be noted that the Levenshtein distance is higher but that the two candidates for the learner realisation are homophonous). As can be seen in our inventory of most frequent retransciptions (Figure 4), some false positives can be observed: *grey/gray* for spelling divergences, *plowing/ploughing*, *sulphur/sulfur* and they account for half of the types of

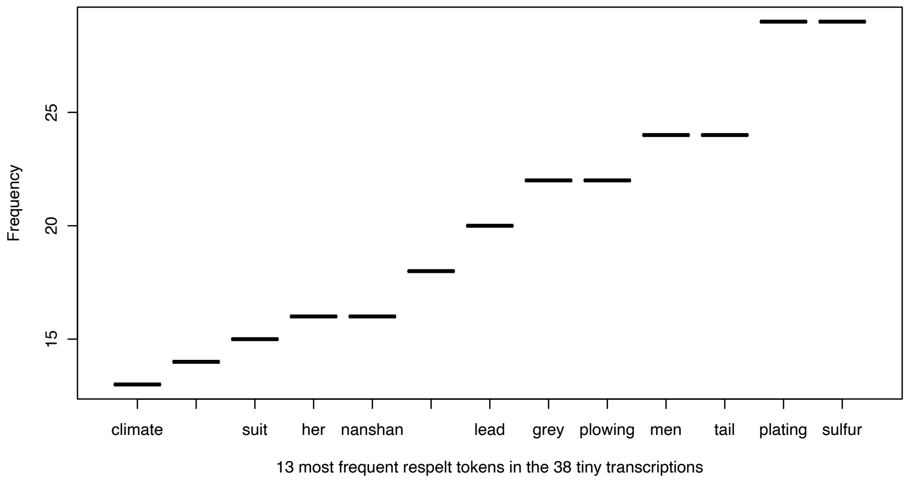


Figure 4: Top 13 aggregated retranscriptions of the `tiny` models

the 13 most frequent respelt tokens. Nevertheless, some frequent retranscriptions actually point to potential misunderstandings by native speakers of the realisations, as they correspond to fully-fledged minimal pairs revealing misrealisations such as *suit* for *soot*.

4.5 Recall of Non-native Phonetic Realisations by Whisper Retranscriptions : The Case of Aspiration

For our analysis of precision of recall, we focused on a very limited number of items in order to evaluate the plausibility of the Whisper retranscriptions, such as *languor* for *language*, and the retranscriptions of the aspiration of *heat* in the sequence *clammy heat*. As to the retranscription of *languor* as *language*, it is mostly due to the realisation as [gw] instead of [g] but the realisation of the final consonant arguably would not trigger misunderstanding for a native speaker. Two experts trained in English phonetics annotated the sound files for limited sequences in relation to our expectations about H-dropping (Exare, 2017). The two experts agreed with the Whisper transcriptions. For our analysis of recall, all the dropped /h/s in *clammy heat* were actually transcribed without an <h>. We auditorily investigated the unaspirated initial /p/ in *pigtail*, which were transcribed as *big*: they were pronounced without initial aspiration. More detailed acoustic analysis of Voice Onset Time (VOT) should be carried out to check the ability of the system to transcribe initial plosives in relation to expected values of Voice Onset Time in English (Abramson and Whalen, 2017; Lisker and Abramson, 1967) and French (Caramazza et al., 1973), in order to investigate whether threshold effects of VOT could be observed in relation to much lower VOT reference values for French (18ms for French /p/ against 59 ms for English).

5 Discussion

At this point, we are not quite able to characterise the different types of “sensitivity" of the Whisper models: tokens do not systematically trigger a spelling variation across all the models.

5.1 Reliability of the detection of error candidates

Some false positives were observed, based on spelling variants (*half-naked* is hyphenated by the Whisper outputs, not in the original). Extra hallucinated ASR errors were observed in the transcriptions in the context of false starts, repairs or repetitions, so that some tokens were repeated several times and occasional cases of coda hallucinations were noticed with *Thank you* or *hit the bell button* being transcribed instead of final silences. Our hypothesis for these cases of coda brittleness of the audio LLM is that part of the training data was initially online and if an end-of-signal cue is captured by the ASR, then this may me transcribed as what might have been left out in the training data.

5.2 Semantically Plausible or Phonetically plausible?

Sequences such as *clammy it* for *clammy heat* raise the question of the semantic plausibility in relation to *surprisal* (Mansfield, 2021), namely the probability of having a given token rather than another one. It seems that the speech inputs, i.e., the phonetic acoustic cues, have more importance in the next-token prediction than just the conditional probability, which would reflect on the semantic plausibility and this apparent dominance of phonetic plausibility seems to prevail over semantic plausibility. Further systematic research analysing surprisal needs to be undertaken, but for an initial estimate of how Whisper outputs may violate semantic plausibility for phonetic plausibility, we computed surprisal using the large language model BERT. We used this to check some of the outputs that were phonetically consistent with the input but semantically less likely: "*He was, however, conscious of being made uncomfortable by the clammy heat. He was, however, conscious of being made uncomfortable by the clammy it*." Even though its surprisal value is much higher with *it* (2.251) compared with the surprisal value for *heat* (0.003), faithfulness to the acoustic signal (absence of aspiration) was observed. This initial foray suggests Whisper outputs are potentially more consistent with phonetic input than with semantic input. In other words, we need to explore the *affordance* (Krunic et al., 2009) of the large language model to accommodate to the acoustic realisations of the learners. How much of the phonetic variability can actually be accommodated by the textual production?

5.3 Alternative Measures of Pronunciation Distance

We did not resort to more elaborated metrics and probably more cognitively grounded measures of pronunciation distance based on the Naive discriminant learning analysis suggested by (Wieling et al., 2014). We are sensitive to the arguments they put forward against Levensthein distance, especially the misalignments produced by the possibility of having reduced vowels. They explain that they have what they call “sensitive sound distances" for tokens like *Wednesday*, which can be realised in a certain number of ways, as two or three syllables. They exemplify the schwa reduction to show that the Levenshtein distance exaggerates the scores in relation to this type of phenomenon. We used the more classical Word Error Rate (WER), which was computed with R (Team, 2023) but we did not apply the normalisation procedure [4] which was used when reporting Whisper performances for WER in (Radford et al., 2022).

5.4 Retranscriptions or Plausible Scenarios for Misunderstanding?

As our examples show, some of the substitutions or respelling proposed by word substitutions and phone substitutions do not necessarily correspond to actual native misunderstandings. In this respect, there is an imbalance between monosyllabic words more likely to convey misunderstanding because of the number of potential minimal pairs (what is known as phonological neighbourhood density) than polysyllabic words, as our *languor* / *language* example seems to suggest. The system is probably biased towards detecting monosyllabic misrealisations more easily, but this also reflects a skewed distribution which can be observed in the language lab exercises, where monosyllabic minimal pairs are much more frequent than polysyllabic examples. Pre-trained generative models are trained to produce tokens, which explains why a word like *funnel* when pronounced initially by a learner as [fju:] becomes transcribed as *funeral*, as this is the closest approximation in spite of the extra syllable.

5.5 Further Validation Procedures

This section discusses potential validation procedures, other than perception tests on native speakers and more detailed acoustic analyses for the transcription of *heat* as *hit*. A list of anticipated phonetic/phonological transfers could potentially be used to serve as the rationale for a confusion matrix analysing the Whisper output and the ability of a graphemic representation to capture phonetic errors.

The ISLE corpus (Menzel et al., 2000; Atwell et al., 2003) has reference transcriptions and validation procedures, but for much shorter segments in carrier sentences such as "*I said wait, not bait*". This corpus of non-native speech also has a read passage by German and Italian speakers, but it has not been annotated by experts. Our preliminary tests with Whisper suggest that heavily-accented speakers are detected as speaking in another language than English and transcribed accordingly.

[4] https://pypi.org/project/whisper-normalizer/

More generally, Whisper has to be tested for other first language speakers, and maybe with other second languages, with the proviso that some languages have a much smaller training size.

5.6 XAI and the Knowledge of the LLM of Different Sizes

Our experiments with Whisper with other recordings suggest that the large-v2 works better, i.e., produces a transcription output which might be more accurate for more sophisticated words. What is the underlying “knowledge" captured in these representations? Is it probably because more data was taken into account in the training phase that "stigmatal and supra-stigmatal features" (`medium` transcription) get (accurately) transcribed as “segmental and supra-segmental features" in the `large-v2` transcriptions. How “linear" is the understanding of the largest models? Is the progression linear between the different transcriptions or can thresholds be observed?

6 Limitations and Further Research

The ASR transcriptions of mispronunciations seem more relevant for segmental features than for supra-segmental features, even though a certain form of chunking is actually captured by a mix of punctuation symbols such as comma and full stops. This means that the word *, however,* in isolation can actually be analysed in terms of successful chunking. Part of the phrasing can be captured by the system through punctuation and, moreover, probably in an even more complex manner, as the end-of-the-line character also of a Whisper transcription corresponds to a form of prosodic chunking different from what is transcribed by a comma or a full stop. In any case, in terms of prosody, only tonality (the ability to properly chunk the prosodic units) can be analysed using Whisper. An important aspect of non-native realisations is the elusive ability to assign stress on the relevant syllables and, in that respect, only reanalyses can be used to track down stress misplacement, as is the case with *her Qulian* for *ˌHercuˈlean*, which is favoured by the stress misplacement. This ASR transcription reveals a weak vowel on *her*, making it more likely to be interpreted as the possessive pronoun. (Kamiyama and Amand, 2023) showed that a frequent incorrect lexical-stress placement amongst French L1 advanced learners of English is the placement of primary stress on the first syllable of words having a similar structure, such as *simulation*, *organisation*. However, unlike the students in Kamiyama & Amand (2023), the students of this study were enrolled in a pronunciation course with a strong focus on stress-imposing endings. The learner whose pronunciation led to the transcription of the form *her Qulian* may have treated the ending *-e.an* like the strong ending *-i.an*, which attracts lexical stress one syllable before the ending, i.e., *Braˈzi.li.an* (Kingdon, 1958).

6.1 Effect of the Training Data on the (Implicit) Rhotic Pronunciation Model

The Librispeech samples available on Hugging face[5] suggest a rather slow reading which is fully rhotic but possibly East coast of the United States (slight variation in the use of yod for *assumed*, *new* or *duke*). There may be a training bias and consequently an implicit rhotic pronunciation model with the data trained on Librispeech (Panayotov et al., 2015). As a baseline for native realisations, we tested the recording of the Librivox version read by Peter Dann, which exhibits a rhotic realisation [6]. The L2 learners of English in this study generally use both rhotic and non rhotic forms while reading the *excerpt from Typhoon*.

6.2 Gender Bias effects

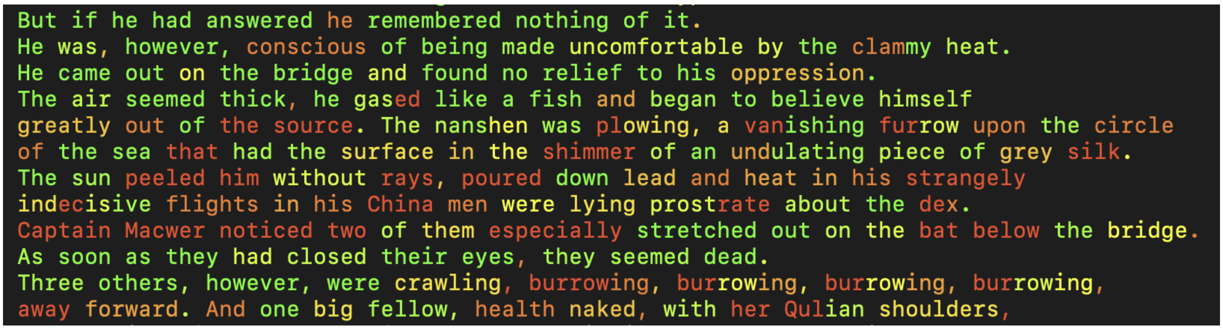
Even though the system revealed that the performances were significantly different for male and female speakers, it is notable that the Levenshtein distances outputted by the `large` model and the `medium` model highlight diverging performances for male and female voices: the `large` model is slightly better than the `medium` for male speakers, but the `medium` model is noticeably better for female speakers.

6.3 Next Steps for ITSs

This subsection discusses how our findings could be implemented in Intelligent Tutoring Systems (ITS). Using an NVIDIA A100 GPU with 40 giga of RAMS, the transcription only took 5 minutes for all the models of two ISLE files, so that the Whisper system could be used to provide almost immediate feedback to learners (or post-hoc analysis when used in a virtual environment). Whisper tran-

[5] `https://huggingface.co/datasets/librispeech_asr`

[6] `ttps://ia802507.us.archive.org/21/items/typhoonandotherstories_2206_librivox/`



```
But if he had answered he remembered nothing of it.
He was, however, conscious of being made uncomfortable by the clammy heat.
He came out on the bridge and found no relief to his oppression.
The air seemed thick, he gased like a fish and began to believe himself
greatly out of the source. The nanshen was plowing, a vanishing furrow upon the circle
of the sea that had the surface in the shimmer of an undulating piece of grey silk.
The sun peeled him without rays, poured down lead and heat in his strangely
indecisive flights in his China men were lying prostrate about the dex.
Captain Macwer noticed two of them especially stretched out on the bat below the bridge.
As soon as they had closed their eyes, they seemed dead.
Three others, however, were crawling, burrowing, burrowing, burrowing, burrowing,
away forward. And one big fellow, health naked, with her Qulian shoulders,
```

Figure 5: Confidence estimation of the predicted tokens as potential visual feedback from *Whisper.cpp* (Gerganov, 2003). Green: confident prediction, i.e., intelligible; red: least confident prediction, i.e., more phonetic training needed to be intelligible. Original text in appendix.

scriptions of non-native speech need to be tested on other tasks than read speech, even though the baseline can be established with the text that was read. With *unscripted*, i.e., spontaneous speech, we may use the `medium` transcription as baseline for the computation of the output of the `tiny` model. In that respect, the existence of several models is an important structural difference from other ASR systems such as *Otter.ai* whose current interface cannot produce a reference text to be compared with Otter's ASR output. For multi-speaker settings such as virtual environments or classroom interactions, speaker *diarisation* will have to be processed first, i.e., the creation of distinct transcribed segments when the speaker changes. Both *Otter.ai* and the experimental C++ implementation of Whisper provide speaker diarisation.

6.4 Scenarios for Potential Visual Feedback

Though experimental, the C++ implementation of Whisper called *Whisper.cpp* (Gerganov, 2003) allows fast processing of some of the Whisper parameters and a visualisation of the confidence estimation for the predicted tokens that is easily understood by teachers and students (Figure 5). The confidence scores are consistent with the phonetic realisations. Stress (mis)placement accounts for some of the scores, as *uncomfortable* was stressed on the penultimate syllable in this example. Running a recording of 151 seconds with its coloured transcription as output only took 4923.62ms on an M1 Pro processor. Feedback can be visually displayed shortly after the end of the recording. For further analyses, a more refined implementation could also output the corresponding confidence scores produced for each subtoken of the transcription (the coloured sequences correspond to the output of byte pair encoding and are not "words").

7 Conclusion

In this paper, we have shown that Whisper's LLM produces different outputs for the transcription task according to the different learner pronunciation models of a reference input. We showed that the number of parameters of the LLM models varied in relation to the detection of tokens varying from the reference text. A phonetic screening of part of the audio files showed the phonetic realism of the retranscriptions varying from the reference file (see appendix). For the analysis of L2 speech, the models trained with fewer parameters paradoxically do a better job at pinpointing L2 pronunciation misrealisations, as they seem more sensitive to phonetic variability than the `large` model. More research is needed to probe the different Whisper models - beyond the model cards (cf. (Mitchell et al., 2019)) that are proposed on the Whisper github [7] - but the analysis of the `tiny` models transcriptions of L2 speech clearly has a future for ICALL systems.

Acknowledgements

This publication has emanated from research supported in part by a 2021 research equipment grant (PAPTAN project)[8] from the Scientific Platforms and Equipment Committee, under the ANR grant (ANR-18-IDEX-0001, Financement IdEx Université de Paris).

[7] https://github.com/openai/whisper/blob/main/model-card.md

[8] Plateforme pour l'apprentissage profond pour la traduction automatique neuronale, in English: Deep Learning for Machine Translation at Universite Paris Cité . See the description of the platform on the project website: https://u-paris.fr/plateforme-paptan

References

Arthur S Abramson and Douglas H Whalen. 2017. Voice Onset Time (VOT) at 50: Theoretical and practical issues in measuring voicing distinctions. *Journal of Phonetics*, 63:75–86.

Eric Atwell, Peter Howarth, and Clive Souter. 2003. The ISLE corpus: Italian and German spoken learner's English. *ICAME Journal: International Computer Archive of Modern and Medieval English Journal*, 27:5–18.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460.

Alfonso Caramazza, Grace H Yeni-Komshian, Edgar B Zurif, and Ettore Carbone. 1973. The acquisition of a new phonological contrast: The case of stop consonants in French-English bilinguals. *The Journal of the Acoustical Society of America*, 54(2):421–428.

Vincent Chanethom and Alice Henderson. 2022. Alignment in ASR and L1 listeners' recognition of L2 learner speech: A replication study. In *15th International Conference on Native and Non-native Accents of English*, Łódź, Poland. Université de Łódź.

George Christodoulides. 2016. *Effects of cognitive load on speech production and perception*. Ph.D. thesis, UCL-Université Catholique de Louvain.

Jonathan Dalby and Diane Kewley-Port. 1999. Explicit pronunciation training using automatic speech recognition technology. *CALICO*, 16(3):425–445.

Christelle Exare. 2017. *Les aspirations intrusives dans l'anglais des apprenants francophones*. Ph.D. thesis, Université Sorbonne Nouvelle.

Christelle Exare. 2022. Awareness of glottal settings for the production of /h/-initial and vowel-initial words in french learners of l2 english. *Anglophonia*, 32.

Georgi Gerganov. 2003. whisper.cpp : A high-performance inference of OpenAI's whisper automatic speech recognition (asr) model.

James Howard II. 2020. Phonetic spelling algorithm implementations for R. *Journal of Statistical Software*, 25(8):1–21.

Mathilde Hutin, Adèle Jatteau, Ioana Vasilescu, Lori Lamel, and Martine Adda-Decker. 2020. Lénition et fortition des occlusives en coda finale dans deux langues romanes : le français et le roumain (lenition and fortition of word-final stops in two Romance languages: French and Romanian). In *Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition), Traitement Automatique des Langues Naturelles (TALN, 27e édition), Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition). Volume 1 : Journées d'Études sur la Parole*, pages 289–298, Nancy, France. ATALA et AFCP.

Chen W. Inceoglu, S. and H. Lim. 2023. Assessment of L2 intelligibility: Comparing L1 listeners and automatic speech recognition. *ReCALL*, 35(1):89–104.

S. Inceoglu, Hyojung Lim, and Wen-Hsin Chen. 2020. ASR for EFL pronunciation practice: Segmental development and learners' beliefs. *The Journal of Asia TEFL*, 17(3):824–840.

Takeki Kamiyama and Maelle Amand. 2023. Perception of word stress amongst French learners of English: nuclear tone suffix. In *Proceedings of the 20th International Congress of Phonetic Sciences, Prague 2023*, ID: 43, pages 2383–2387.

Roger Kingdon. 1958. *The Groundwork of English stress*. Longmans.

V. Krunic, G. Salvi, A. Bernardino, L. Montesano, and J. Santos-Victor. 2009. Affordance based word-to-meaning association. In *2009 IEEE International Conference on Robotics and Automation*, pages 4138–4143.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Leigh Lisker and Arthur S Abramson. 1967. Some effects of context on voice onset time in English stops. *Language and speech*, 10(1):1–28.

Courtney Mansfield. 2021. *ASR and human recognition errors: Predictability and lexical factors*. Ph.D. thesis, University of Washington.

Wolfgang Menzel, Eric Atwell, Patrizia Bonaventura, Daniel Herron, Peter Howarth, Rachel Morton, and Clive Souter. 2000. The ISLE corpus of non-native spoken English. In *Proceedings of LREC 2000: Language Resources and Evaluation Conference, vol. 2*, pages 957–964. European Language Resources Association.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. ACM.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an ASR corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5206–5210. IEEE.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The Kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, CONF. IEEE Signal Processing Society.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision.

C.L. Rogers, J. M. Dalby, and G. DeVane. 1994. Intelligibility training for foreign-accented speech: A preliminary study. *JASA*, 96(5):3348.

R Core Team. 2023. R: A language and environment for statistical computing.

Steven Weinberger. 2015. Speech accent archive. George Mason University. *http://accent.gmu.edu*.

Martijn Wieling, John Nerbonne, Jelke Bloem, Charlotte Gooskens, Wilbert Heeringa, and R Harald Baayen. 2014. A cognitively grounded measure of pronunciation distance. *PloS one*, 9(1):e75734.

Appendix

Observing the steady fall of the barometer, Captain MacWhirr thought, "There's some dirty weather knocking about." This is precisely what he thought. He had had an experience of moderately dirty weather—the term dirty as applied to the weather implying only moderate discomfort to the seaman. Had he been informed by an indisputable authority that the end of the world was to be finally accomplished by a catastrophic disturbance of the atmosphere, he would have assimilated the information under the simple idea of dirty weather, and no other, because he had no experience of cataclysms, and belief does not necessarily imply comprehension. The wisdom of his county had pronounced by means of an Act of Parliament that before he could be considered as fit to take charge of a ship he should be able to answer certain simple questions on the subject of circular storms such as hurricanes, cyclones, typhoons; and apparently he had answered them, since he was now in command of the Nan-Shan in the China seas during the season of typhoons. But if he had answered he remembered nothing of it. He was, however, conscious of being made uncomfortable by the clammy heat. He came out on the bridge, and found no relief to this oppression. The air seemed thick. He gasped like a fish, and began to believe himself greatly out of sorts.

The Nan-Shan was ploughing a vanishing furrow upon the circle of the sea that had the surface and the shimmer of an undulating piece of gray silk. The sun, pale and without rays, poured down leaden heat in a strangely indecisive light, and the Chinamen were lying prostrate about the decks. Their bloodless, pinched, yellow faces were like the faces of bilious invalids. Captain MacWhirr noticed two of them especially, stretched out on their backs below the bridge. As soon as they had closed their eyes they seemed dead. Three others, however, were quarrelling barbarously away forward; and one big fellow, half naked, with herculean shoulders, was hanging limply over a winch; another, sitting on the deck, his knees up and his head drooping sideways in a girlish attitude, was plaiting his pigtail with infinite languor depicted in his whole person and in the very movement of his fingers. The smoke struggled with difficulty out of the funnel, and instead of streaming away spread itself out like an infernal sort of cloud, smelling of sulphur and raining soot all over the decks.

Enhancing Word Discrimination and Matching in Query-by-Example Spoken term detection with Acoustic Word Embeddings

Pantid Chantangphol and **Theerat Sakdejayont** and **Tawunrat Chalothorn**
Kasikorn Labs Kasikorn Business–Technology Group, Thailand
{pantid.c,theerat.s,tawunrat.c}@kbtg.tech

Abstract

In this paper, we propose a novel approach to enhance query-by-example spoken term detection using Acoustic Word Embeddings (AWEs). Our AWEs model combines CNN and LSTM layers to capture sequential information and generate fixed-dimensional word-level embeddings. To address the challenge of distinguishing between words, we introduce a deep word discrimination loss that enhances embedding discrimination. Additionally, we employ an embedding-matching scheme based on cosine similarity computation and sliding window smoothing. Our experimental results demonstrate the effectiveness of our approach in word discrimination tasks, achieving high mean Average Precision scores and outperforming baseline models. Moreover, our embedding-matching scheme shows promising performance in query-by-example spoken term detection, opening up possibilities for advancements in audio indexing and search techniques.

Index Terms: spoken term detection, query-by-example, acoustic word embedding, word discrimination, audio retrieval

1 Introduction

The field of Spoken Term Detection (STD) (Mandal et al., 2014)—identifying specific terms within audio streams or files—has gained importance due to the widespread availability of internet media and the proliferation of smart devices. This has led to an increasing demand for proficient audio search tools and efficient voice control mechanisms. Query by Example (QbE) represents a specialized application of STD, offering advantages over traditional text-based searches by directly matching audio samples. This is especially valuable for handling unknown or out-of-vocabulary search terms.

Query by Example Spoken Term Detection (QbE-STD) has historically employed Dynamic Time Warping (DTW) in conjunction with frame-level features for keyword matching (Rodriguez-Fuentes et al., 2014; Mantena et al., 2014). Both supervised (Zhang et al., 2019) and unsupervised approaches (Chen et al., 2016; Holzenberger et al., 2018) have been examined, each with distinct advantages. While unsupervised methods primarily utilize traditional acoustic features (Vasudev et al., 2016; Wang et al., 2018), supervised techniques frequently employ neural network-derived phonetic features. The field has witnessed a paradigm shift with the introduction of Acoustic Word Embeddings (AWEs) (Ma et al., 2021; Kamper et al., 2019; Settle et al., 2017; Kamper et al., 2016; Yuan et al., 2018), which transform variable-length speech segments into fixed-dimensional vectors (Levin et al., 2013). This approach overcomes the computational limitations of traditional DTW-based methods, facilitating more efficient searching, clustering, and similarity comparisons. Neural networks, particularly Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks, are now widely utilized for extracting these AWEs (Ram et al., 2018; Svec et al., 2022; Chen et al., 2015; Settle and Livescu, 2016; Chung and Glass, 2018; Naik et al., 2020; Ram et al., 2020; Lopez-Otero et al., 2019; Madhavi and Patil, 2017). Consequently, the current focus in QbE-STD research has largely shifted towards search and indexing tasks, with these deep learning frameworks playing a pivotal role in feature extraction.

The main challenge resides in mapping sequential speech information into vector space without losing sequential integrity. Our proposed method addresses this challenge through deep neural networks and introduces an additional loss function designed for enhanced word discrimination. This paper presents an architecture combining CNN layers for local feature extraction, Long Short-Term Memory (LSTM) layers for capturing temporal dependencies, and Fully Connected Layers (FC

Layers) for dimensionality reduction. An additional loss function is incorporated to improve word discrimination and optimize generalization across keywords spoken by various speakers by aligning embeddings with acoustic word centroids while maximizing inter-class and minimizing intra-class variation. Moreover, our method utilizes a cosine similarity-based query centroid matching technique, supplemented by moving average smoothing, for efficient word search in spoken utterances. Our contributions to this work are as follows:

1. Introduction of an Acoustic-to-Embedding network (A2E-Net) for generating word-level acoustic word representations.

2. Development of a Deep Word Discrimination (DWD) loss function aimed at enhancing the discrimination capabilities of acoustic word embeddings by minimizing intra-word distance and maximizing inter-word distance within each acoustic word embedding.

3. Establishment of Query Centroid Similarity Matching (QC-matching), a technique for acoustic word embedding matching that employs query centroids to facilitate QbE-based audio indexing.

The remainder of this paper is organized as follows: Section 2 details the proposed system, Section 3 discusses implementation aspects, Section 4 presents the results, and Section 6 outlines the conclusions.

2 Proposed framework

In this section, we present the components of our proposed method for enhancing QbE-STD. They are as follows:

2.1 Acoustic-to-Embedding network (A2E-Net)

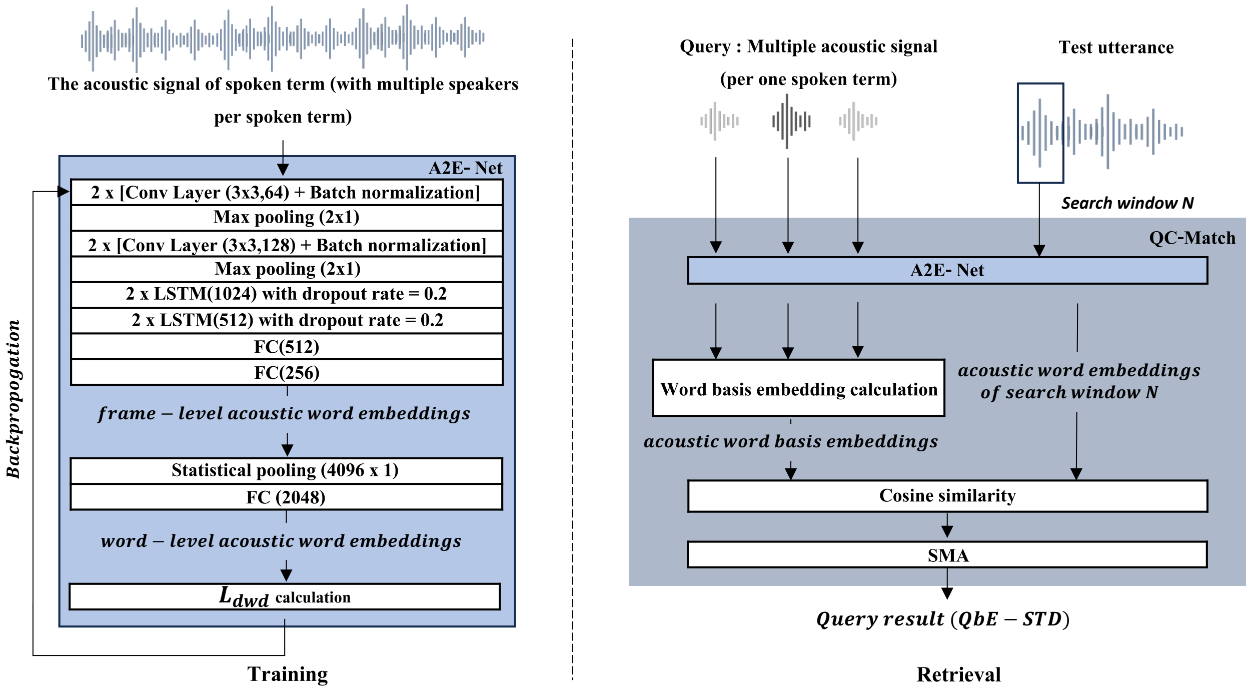
Our proposed AWEs model architecture aims to effectively capture and represent acoustic features at both the frame and word levels. The input comprises raw audio signals, which are divided into frames using a windowing size of 25 ms and a step size of 10 ms. To extract local acoustic features, we employ two CNN layers with 3x3 kernels and 64 filters each, followed by a max-pooling layer that reduces dimensions and extracts essential features. Two additional CNN layers with 3x3 kernels and 128 filters each extract higher-level features, followed by another max-pooling layer for further dimension reduction.

To capture temporal dependencies and sequence information, we utilize two sets of LSTM layers. The first set consists of two LSTM layers with 1024 units, followed by another set of two LSTM layers with 512 units each. These LSTM layers are crucial for modeling the sequential nature of acoustic features. Subsequently, two FC layers map the LSTM outputs to lower-dimensional spaces, reducing dimensionality and facilitating subsequent embeddings. The resulting frame-level AWEs, with a size of 256x1, are obtained from the output of the FC layer. The statistical pooling layer then aggregates the variable-length frame-level AWEs into a fixed-length representation by computing the mean and standard deviation, concatenating these, and finally mapping them to a 4096-dimensional space through a linear transformation. This fixed-length representation encapsulates both the mean and variance of the frame-level features, making it a rich and comprehensive descriptor for each word. Another FC layer maps a 4096x1 representation to a 2048-dimensional space, generating word-level AWEs. During training, the model parameters are optimized using both cross-entropy loss, a common classification loss, and an auxiliary word-discrimination loss designed to enhance embedding discrimination.

In summary, our AWEs model architecture combines CNN layers for local feature extraction, LSTM layers for capturing temporal dependencies, and FC layers for dimensionality reduction and mapping to lower-dimensional embeddings. By representing acoustic features at both the frame and word levels, our model enables the effective calculation of word-level embeddings and facilitates meaningful similarity comparisons.

2.2 Deep word discrimination Loss (DWD)

The DWD loss is introduced to address the challenge of accurate word discrimination. In such tasks, where the search content and query keyword are typically spoken by different speakers, it is crucial to ensure that the AWEs of the same spoken keyword by different speakers are identical. However, traditional embedding approaches often encode speaker-related information, which hinders precise word discrimination. To overcome this limitation, we incorporate a variability-invariant loss in the training phase.



The acoustic signal of spoken term (with multiple speakers per spoken term)
A2E- Net
2 x [Conv Layer (3x3,64) + Batch normalization]
Max pooling (2x1)
2 x [Conv Layer (3x3,128) + Batch normalization]
Max pooling (2x1)
2 x LSTM(1024) with dropout rate = 0.2
2 x LSTM(512) with dropout rate = 0.2
FC(512)
FC(256)
frame – level acoustic word embeddings
Statistical pooling (4096 x 1)
FC (2048)
word – level acoustic word embeddings
L_{dwd} calculation
Backpropogation
Training
Query : Multiple acoustic signal (per one spoken term)
Test utterance
Search window N
QC-Match
A2E- Net
Word basis embedding calculation
acoustic word embeddings of search window N
acoustic word basis embeddings
Cosine similarity
SMA
Query result (QbE – STD)
Retrieval

Figure 1: A2E-net with QC-matching framework for Query-by-Example Spoken term detection system

To address the inter-class and intra-class covariance in QbE-STD tasks, our additional loss function aims to maximize variation across different word classes while minimizing variation within the same class. We construct batches of size $N_{word} \times M$, where M denotes the number of acoustic signals for each word, spoken by different speakers. This is designed to capture the diversity in pronunciation, accent, and other speech characteristics unique to each speaker. Such diversity is crucial in a QbE-STD task, where the goal is to accurately identify a keyword regardless of the speaker. By incorporating acoustic signals from multiple speakers for each word, the model is trained to recognize words independently of speaker-specific characteristics. N_{word} represents the number of distinct words and indicates the size of the vocabulary in the training set. This set is generated from the alignment of acoustic signals and their transcriptions using the Montreal Forced Aligner (McAuliffe et al., 2017). After alignment, feature vectors x_{ji} are extracted from the i^{th} acoustic signal of the j^{th} word. These vectors are input into the AWEs model, comprising convolutional layers with ReLU activation and batch normalization, followed by max-pooling layers. The model also includes four LSTM layers and two dense layers with ReLU activation, culminating in the output layer that represents frame-level AWEs. Each word-level AWE emb_{ji} is normalized to enable accurate comparisons.

The main objective during training is to optimize the embedding representation of each acoustic signal. This involves aligning the embedding closely with the centroid of embeddings from the same word while ensuring a significant separation from centroids of other words. The word embedding centroid is computed by averaging the word-level embeddings, excluding the i^{th} acoustic word embedding, denoted as $[emb_{j1}, ..., emb_{jM}]$ with the M acoustic signals per word, resulting in c_j.

$$c_j = \frac{\sum_{m=1,m\neq i}^{M} emb_{jm}}{M-1} \tag{1}$$

To measure the similarity between the word-level embeddings and the centroid, we employ cosine similarity. The similarity matrix $(S_{ji,k})$ represents the scaled cosine similarities between each embedding vector emb_{ji} and all centroids c_k.

$$S_{ji,k} = \cos\left(emb_{ji}, c_k\right) \tag{2}$$

To enhance conventional contrastive loss in QbE-STD tasks, a softmax operation is applied to similarity scores, enabling a probabilistic interpretation of the similarity between embedding vectors. The loss on each embedding vector (emb_{ji}) is defined as follows:

$$L_{sm} = -S_{ji,j} + \log \sum_{k=1}^{N_{word}} \exp S_{ji,k} \quad (3)$$

where L_{sm} represents the softmax loss.

Finally, we introduce the contrastive centroid Loss (L_{cc}) to encourage embeddings of positive examples (words in the query) to be close to their respective class centers while simultaneously pushing them away from the class centers of negative examples (other words). By considering both the centrality and contrastive aspects, this loss promotes effective discrimination in QbE audio indexing.

$$L_{cc} = \sum_{j=1}^{N_{word}} \sum_{i=1}^{M} (1 - S_{ji,j}) + \max_{1<k<N_{word}, j \neq k} S_{ji,k} \quad (4)$$

The $(1 - S_{ji,j})$ targets positive pairs, measuring and minimizing their dissimilarity from the class center to enhance intra-class compactness. The second term addresses the most dissimilar negative pairs. It identifies the maximum similarity between emb_{ji} and centroids of all other classes $(k \neq j)$ The aim is to decrease the similarity of an embedding vector to centroids of different words, thus increasing inter-class variability.

The Deep Word Discrimination loss (L_{dwd}) is a combination of the softmax loss and the contrastive centroid Loss, as follows:

$$L_{dwd} = L_{sm} + L_{cc} \quad (5)$$

By incorporating the Deep Word Discrimination loss into the training process, our goal is to enhance the discriminative power of the embeddings, thereby facilitating accurate word discrimination in QbE search tasks.

2.3 Query centroid similarity matching (QC-matching)

Our proposed word-searching system employs an embedding-matching scheme based on cosine similarity computation with a sliding window. To initiate the process, the search content is divided into segments using a fixed-size sliding window along the time axis, forming a sequence of segments. These segments are then passed through a trained A2E-Net, resulting in a sequence of acoustic word embeddings derived from the FC layer.

To ensure consistency in segment lengths, the keyword audio is either padded or clipped to match the size of the sliding window. Subsequently, each input segment (x) is transformed into its corresponding embedding (emb_x) using deep CNN. In order to capture the representation of acoustic signals of a spoken query term, the basis embedding of the word is computed by averaging the word-level embeddings in the following manner:

$$c_b = \frac{\sum_{x=1}^{B} emb_x}{B} \quad (6)$$

where B is the number of multiple acoustic signals of a spoken query term. The basis embedding, denoted as c_b, captures the representative acoustic features of the spoken query.

By calculating the cosine similarity between the segment sequence of the search content y and the basis embedding of the spoken query (emb_x), we generate a time-dependent score sequence. To mitigate the impact of random score fluctuations, we apply a simple moving average (SMA) operation (Koul and Awasthi, 2019) to smooth the sequence. This smoothing process involves summing recent scores and dividing the sum by the number of frames involved at each point.

The resulting smoothed score sequence provides a measure of similarity between the search content and the spoken query, enabling the identification of relevant word occurrences within the search content. This embedding-matching approach, employing cosine similarity computation with a sliding window and subsequent SMA smoothing, offers an effective means of searching for specific words in spoken utterances.

3 Experimental Details

In this section, we provide the experimental details of our study, covering evaluation metrics, the dataset, baselines, data preparation, and model configuration.

3.1 Evaluation

Our evaluation of the method employs two key metrics: mean Average Precision (mAP) and Precision at 5 (P@5), same as (Ma et al., 2021). The mAP metric assesses the average precision for each word in word discrimination and search content. It is calculated by averaging precision values for all queries, providing a holistic measure of retrieval performance. Precision at k documents (P@k) evaluates the precision of the retrieval system by considering the relevance of the top k retrieved word occurrences. By using mAP and P@5, we gain insights into the retrieval performance and precision of our word-searching system, accurately retrieving desired words from spoken utterances.

3.2 Dataset

This study explores word discrimination across Buckeye (Pitt et al., 2005) (6 hours for development and testing), Librispeech (Panayotov et al., 2015) (5.4 hours for development and clean testing), TIMIT (Garofolo et al., 1983) (4620 audio files for training, 1690 for testing), and English Command Voice corpus 12.0 (Ardila et al., 2020) (986,897 utterances for training, 16,365 for development and testing).

To evaluate word discrimination, we train an AWEs model using the English Common Voice dataset and assess discrimination using Librispeech and Buckeye. We investigate the QbE technique for spoken term detection and compare the performance of our embedding-matching method with other approaches. For embedding-matching, we use spoken queries from Librispeech and test utterances from TIMIT. We examine the effectiveness of fixed-dimensional acoustic embedding by obtaining unseen spoken queries from Librispeech and test utterances from TIMIT. Through these experiments, our aim is to gain insights into word discrimination and evaluate the effectiveness of our proposed method in unseen word search scenarios.

3.3 Baseline

Network: Due to the high performance of supervised acoustic word embedding models, as cited in (Ram and Aldarmaki, 2022) and (Sanabria et al., 2023), we evaluate our proposed AWE model in comparison with baseline models such as Wav2Vec 2.0 (W2V2) (Baevski et al., 2020), HuBERT (Hsu et al., 2021), and XLSR-53 (Conneau et al., 2021) for word discrimination tasks. These baseline models leverage pre-trained supervised representations for constructing acoustic word embeddings (AWEs). Notably, HuBERT with mean-pooling outperforms other AWE systems employing simpler pooling strategies, as evidenced in (Sanabria et al., 2023), thus showcasing its robust performance across various AWEs. Additionally, XLSR-53 demonstrates promising performance, as reported in (Ram and Aldarmaki, 2022).

Loss function: We compare the performance of our DWD loss with other methods, including Triplet loss (Ge et al., 2018) and Multi-Similarity Loss (MS loss) (Wang et al., 2019), demonstrating the strong and consistent performance of our DWD loss across various AWEs.

Word matching : Furthermore, we compare our proposed embedding-matching approach for QbE-STD with the baseline Cosine Distance Pattern Matching (CDP matching) method (Ma et al., 2021). This baseline method employs cosine distance computation in conjunction with a sliding window to match spoken query segments to the search content. A simple moving average is then applied to smooth the score sequence, thereby reducing random fluctuations. Additionally, a multi-template strategy is used to average values across templates, resulting in a fused embedding. We also compare our proposed approach with the best-performing model, One Softmax AWE with V-I Loss (s-AWE), as outlined in the work of (Ma et al., 2021).

3.4 Experimental setup

To evaluate the performance of A2E-Net in word discrimination tasks, reference is made to the experiment detailed in Section 4.2. We utilize six different systems for this evaluation: the proposed A2E-Net with DWD loss, softmax loss, and MS loss, as well as pre-trained W2V2, Hubert, and XLSR-53 models. The objective is to examine the efficacy of A2E-Net across different loss functions, including DWD, softmax, and MS loss. Performance comparisons are made against high-performing pre-trained models. The metric for evaluation is mAP, and word categorization employs a cosine similarity threshold of 0.5.

To assess the proposed Query-by-Example (QbE) approach for Spoken Term Detection (STD), experiments outlined in Section 4.2 are referenced. The systems examined specifically include A2E-Net with QC matching, A2E-Net with CDP matching, and s-AWE with CDP matching (Ma et al., 2021). The performance of A2E-Net is scrutinized by employing various word-matching methods and is compared against the benchmark technique of CDP matching.

To evaluate the efficacy of the proposed method in word discrimination tasks, an experiment was conducted to compare frame-level and word-level acoustic embeddings. Two variations of the A2E-Net model were employed: one with DWD loss and another with softmax loss. Detailed results and analyses can be found in Section 4.3.

To investigate the effectiveness of the proposed method in retrieving unseen words for real-world applications, an experiment was conducted as outlined in Section 4.4. The word-level A2E-Net with DWD loss was used, and the experiment focused on two categories of words: all-selected and unseen. The objective is to evaluate the ability of the system to retrieve and rank unseen words compared to a pre-selected set of words.

3.5 Data Preparation

Speech signals in all experiments were processed at a sampling rate of 16 kHz with 16-bit resolution.

To train the word discrimination model and conduct evaluations, precise word timestamps were necessary. Forced alignment techniques were employed for datasets without manual timestamps, using the MFA (McAuliffe et al., 2017) for all datasets. The evaluation focused on words with a minimum duration of 0.5 seconds.

During word discrimination training and inference, acoustic word segments were divided into 25 ms frames with a step size of 10 ms. These frames were transformed into 25-dimensional feature vectors for the acoustic word embedding model. The model generated embeddings, and cosine similarity with a threshold was used for comparison and classification.

For embedding-matching in QbE-STD, a query word with multiple acoustic words was indexed within a recording file. The word basis embedding and average duration of the query word were calculated. The recording file was segmented into segments of the average duration with a step size of 50 ms. Acoustic word embeddings were compared to the word basis embedding using cosine similarity, enabling identification and indexing based on a similarity threshold.

3.6 Model Configuration

To compare with the baseline, we conducted experiments using frame-level and word-level representations from various models. For frame-level representations, we evaluated word discrimination models with different loss functions. For word-level representations, we examined word discrimination models with the DWD loss.

For each reported model, we employed specific hyperparameter configurations, including a learning rate of 0.001, a batch size of 32, and the Adam optimizer. The output layer of the word discrimination model generated a 2048-dimensional word embedding with N_{word} nodes, representing the number of words in the training set. We implemented early stopping, and halting training if the validation loss did not improve for more than 10 epochs or started to increase for more than 3 epochs. The maximum number of epochs was set to 100. These hyperparameter settings and training strategies played a crucial role in achieving optimal model performance.

Table 1: The performance evaluation of A2E-Net in word discrimination task

Methods		mAP(%)	
Model	Loss	Librispeech	Buckeye
A2E-Net	DWD loss	**63.9**	**72.9**
	softmax loss	59.1	65.2
	Triplet loss	60.2	68.3
	MS loss	62.5	69.1
W2V2 (Baseline)		47.4	53.1
Hubert (Baseline)		58.2	64.8
XLSR-53 (Baseline)		54.7	60.1

4 Experimental result and discussion

In this section, we present the experimental results and discussion of our study, focusing on performance evaluation and comparisons across various aspects.

4.1 The performance evaluation of A2E-Net in word discrimination task

This study investigates the performance of various model architectures in word discrimination tasks using our proposed method. In Table 1, we compare the effectiveness of the A2E-Net model across different loss functions (DWD, softmax, Triplet, and MS) against two baseline models (W2V2 and HuBERT), employing the mAP metric for evaluation. These results contribute to the advancement of word embedding models. Specifically, the A2E-Net model with DWD loss demonstrates exceptional performance, achieving the highest mAP scores of 63.9% for Librispeech and 72.9% for Buckeye, thus outperforming both baseline models. Furthermore, the A2E-Net model employing the softmax loss function also shows competitive performance, with mAP scores of 59.1% for Librispeech and 65.2% for Buckeye. However, there remains room for further optimization. In contrast, W2V2 exhibits moderate performance, and although HuBERT outperforms W2V2, it still falls short of the mAP scores achieved by the A2E-Net

Table 2: The performance evaluation of a proposed QbE Approach for QbE-STD

Methods		mAP (%)	P@5 (%)
Model	Word matching		
A2E-Net	QC-matching	**70.22**	**80.62**
A2E-Net	CDP matching	59.1	65.2
Baseline			
s-AWE	CDP matching	59.1	65.2

models. The XLSR-53 model also demonstrates promise but requires additional tuning to match the performance of our proposed models. Overall, the A2E-Net model with the DWD loss function emerges as the most effective architecture for word discrimination tasks, highlighting the efficacy of its design and chosen loss function in achieving superior performance. This research offers valuable insights into various model architectures for word discrimination, thereby guiding future investigations in this field.

4.2 The performance evaluation of a proposed QbE Approach for QbE-STD

We conducted a comprehensive investigation to assess the effectiveness of our QbE technique for QbE-STD in Table 2, comparing it with existing approaches. We implemented two variations of our model: word embedding-based matching with a proposed loss function and pattern matching based on cosine distance with the same loss function. The evaluation was performed using the mAP metric, and the results were compared to baseline approaches. The word embedding-based model achieved high mAP scores of 70.22%, effectively detecting spoken terms. On the other hand, the pattern matching-based model showed strengths in capturing patterns but exhibited slightly lower performance. In contrast, the baseline models had lower mAP scores, indicating limitations in STD. Ultimately, the word embedding-based model emerged as the most effective, outperforming the baseline models. Our findings highlight the potential of QbE techniques and pave the way for future improvements in STD methods.

4.3 The performance evaluation of frame-level and word-level Acoustic Word Embeddings for word discrimination task

This experiment evaluates various architectures for word discrimination tasks using frame-level and word-level acoustic word embeddings. Our proposed model, which employs a specialized loss function, is compared with its softmax loss variant using the mean Average Precision (mAP) metric. Results in Figure 2 show that the specialized loss function yields high mAP scores for both frame-level and word-level embeddings, highlighting its efficacy in word discrimination. Moreover, word-level representation outperforms its frame-level counterpart, capturing discrimination patterns more effectively. The proposed model also surpasses the softmax model, validating the effectiveness of our architecture and loss function. In conclusion, we recommend using the word-level approach with our specialized loss function to improve word discrimination models, contributing to advances in speech analysis.

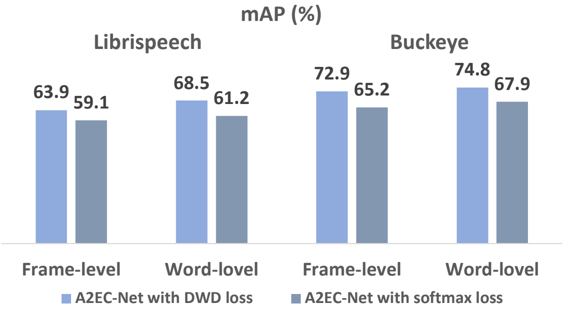


Figure 2: The performance evaluation of frame-level and word-level Acoustic Word Embedding for word discrimination task

Table 3: Performance Evaluation of AWEs for Unseen Word Retrieval

Retrieval	mAP (%)	P@5 (%)
All selected words	70.22	80.62
Unseen words	54.95	62.59

4.4 The performance evaluation of AWEs for Unseen Word Retrieval

This experiment evaluates the effectiveness of AWEs in retrieving unseen words through QC-matching, utilizing A2E-Net and the DWD loss. We measure the system's performance in identifying and retrieving unseen words compared to randomly selected words, using the mAP metric. The results presented in Table 3 advance search techniques for speech data, offering valuable insights into the effectiveness of AWEs in Unseen Word Search Retrieval. By analyzing the strengths and weaknesses of each architecture in word discrimination tasks, the proposed model with AWEs demonstrates impressive performance in distinguishing both seen and unseen words across languages. It achieves high mAP and P@5 scores, although there is still room for improvement in discriminating unseen words. These findings highlight the effectiveness of AWEs for word discrimination and emphasize the benefits of leveraging multilingual models. Overall, they contribute to the advancement of search techniques for STD, providing valuable insights for future research in this domain.

5 Discussion

5.1 Performance Insights

Acoustic Word Embeddings (AWEs) have played a pivotal role in advancing the field by offering a computationally efficient approach to spoken term detection. Our A2E-Net model with DWD loss function outperformed baseline models like W2V2 and HuBERT, achieving mAP scores of 63.9% on Librispeech and 72.9% on Buckeye. These scores underline the architectural efficiency and the efficacy of the DWD loss function. On both frame-level and word-level tasks, our specialized loss function improves word discrimination, thereby enhancing the versatility of the model across different granularities. AWEs were also effective in retrieving unseen words, thereby advancing search techniques for speech data. Our QbE technique surpassed existing baseline models with a high mAP score of 70.22%, underscoring the efficacy of word embedding-based models in spoken term detection.

5.2 Computation Time

One notable advantage of A2E-Net is its computational efficiency. Traditional methods (e.g. DTW) suffer from high computational complexity, especially with long sequences. A2E-Net generates AWEs that represent variable-length segments as fixed-dimensional vectors, significantly reducing computation time for search and similarity comparisons. While the training phase is resource-intensive due to the depth of the model, real-time deployment remains efficient. The specialized loss function adds minimal computational overhead, making model scalable for real-time applications.

5.3 Theoretical and Practical Implications

The research findings have important theoretical ramifications for the academic community in machine learning, acoustic modeling, and natural language processing. On the practical side, the reduced computational complexity and time efficiencies hold promise for applications in information retrieval, speech indexing, and automated customer service.

5.4 Limitations and Future Work

Despite encouraging results, limitations exist. The proposed loss functions, though superior to traditional ones, require broader linguistic testing. Additional evaluation against a more diverse set of baseline models could enrich our findings. The current A2E-Net model excels in distinguishing seen words but falls short in discriminating unseen words. Future work could focus on developing adaptive methods to enhance this specific performance aspect. The generalizability of the model across various languages, dialects, or noisy environments, as well as its practical effectiveness in real-world, real-time applications, remains to be tested. Moreover, subsequent studies could expand the A2E-Net model to include more languages, particularly those with limited resources, to increase its applicability in linguistically diverse contexts. Therefore, upcoming research could focus on overcoming these limitations and further refining the performance of the model across multiple domains.

6 Conclusion

The presented research substantially advances the understanding and development of Query-by-Example Spoken Term Detection (QbE-STD) techniques, acoustic word embeddings (AWEs), and their integration with deep learning architectures. Our study introduces an innovative approach to enhance QbE-STD through the use of AWEs. The A2E model overcomes the limitations of traditional methods by converting variable-length speech segments into fixed-dimensional vectors, thereby facilitating quicker and more efficient search operations. Experimental results confirm the model's effectiveness in word discrimination tasks, underscoring its potential for innovations in audio indexing and search techniques. The incorporation of the DWD loss function further augments the discriminative power of the embeddings. Our contributions not only advance the field of QbE-STD but also set the stage for improved audio search tools and voice-controlled applications. Particularly, the A2E-Net model with DWD loss function exhibits superior performance, offering promising avenues for future research in speech technology.

References

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France. European Language Resources Association.

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. Wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Guoguo Chen, Carolina Parada, and Tara N. Sainath. 2015. Query-by-example keyword spotting using long short-term memory networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5236–5240.

Hongjie Chen, Cheung-Chi Leung, Lei Xie, Bin Ma, and Haizhou Li. 2016. Unsupervised bottleneck features for low-resource query-by-example spoken term detection. In *17th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 923–927. ISCA.

Yu-An Chung and James R. Glass. 2018. Speech2vec: A sequence-to-sequence framework for learning word embeddings from speech. In *19th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 811–815. ISCA.

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. 2021. Unsupervised cross-lingual representation learning for speech recognition. In *22nd Annual Conference of the International Speech Communication Association (Interspeech)*, pages 2426–2430. ISCA.

John S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathan G. Fiscus, David S. Pallett, Nancy L. Dahlgren, and Victor Zue. 1983. Timit acoustic-phonetic continuous speech corpus.

Weifeng Ge, Weilin Huang, Dengke Dong, and Matthew R. Scott. 2018. Deep metric learning with hierarchical triplet loss. In *Computer Vision – ECCV 2018*, pages 272–288, Cham. Springer International Publishing.

Nils Holzenberger, Mingxing Du, Julien Karadayi, Rachid Riad, and Emmanuel Dupoux. 2018. Learning word embeddings: Unsupervised methods for fixed-size representations of variable-length speech segments. In *19th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 2683–2687. ISCA.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.

Herman Kamper, Aristotelis Anastassiou, and Karen Livescu. 2019. Semantic query-by-example speech search using visual grounding. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7120–7124.

Herman Kamper, Weiran Wang, and Karen Livescu. 2016. Deep convolutional acoustic word embeddings using word-pair side information. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4950–4954.

Sumit Koul and Amit Kumar Awasthi. 2019. An introduction to moving average and its importance. *Journal of emerging technologies and innovative research*.

Keith Levin, Katharine Henry, Aren Jansen, and Karen Livescu. 2013. Fixed-dimensional acoustic embeddings of variable-length segments in low-resource settings. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 410–415.

Paula Lopez-Otero, Javier Parapar, and Alvaro Barreiro. 2019. Efficient query-by-example spoken document retrieval combining phone multigram representation and dynamic time warping. *Information Processing and Management*, 56(1):43–60.

Murong Ma, Haiwei Wu, Xuyang Wang, Lin Yang, Junjie Wang, and Ming Li. 2021. Acoustic word embedding system for code-switching query-by-example spoken term detection. In *2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 1–5.

Maulik C. Madhavi and Hemant A. Patil. 2017. Partial matching and search space reduction for qbe-std. *Computer Speech and Language*, 45:58–82.

Anupam Mandal, K. R. Prasanna Kumar, and Pabitra Mitra. 2014. Recent developments in spoken term detection: a survey. *International Journal of Speech Technology*, 17(2):183–198.

Gautam Mantena, Sivanand Achanta, and Kishore Prahallad. 2014. Query-by-example spoken term detection using frequency domain linear prediction and non-segmental dynamic time warping. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(5):946–955.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *18th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 498–502. ISCA.

Prajyot Naik, Manisha Naik Gaonkar, Veena Thenkanidiyoor, and A. D. Dileep. 2020. Kernel based matching and a novel training approach for cnn-based qbe-std. In *2020 International Conference on Signal Processing and Communications (SPCOM)*, pages 1–5.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Mark A. Pitt, Keith Johnson, Elizabeth Hume, Scott Kiesling, and William Raymond. 2005. The buckeye corpus of conversational speech: labeling conventions and a test of transcriber reliability. *Speech Communication*, 45(1):89–95.

Dhananjay Ram, Lesly Miculicich, and Hervé Bourlard. 2018. CNN based query by example spoken term detection. In *19th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 92–96. ISCA.

Dhananjay Ram, Lesly Miculicich, and Hervé Bourlard. 2020. Neural network based end-to-end query by example spoken term detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1416–1427.

Sreepratha Ram and Hanan Aldarmaki. 2022. Supervised acoustic embeddings and their transferability across languages. In *Proceedings of the 5th International Conference on Natural Language and Speech Processing (ICNLSP 2022)*, pages 212–218, Trento, Italy. Association for Computational Linguistics.

Luis J. Rodriguez-Fuentes, Amparo Varona, Mikel Penagarikano, Germán Bordel, and Mireia Diez. 2014. High-performance query-by-example spoken term detection on the sws 2013 evaluation. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7819–7823.

Ramon Sanabria, Hao Tang, and Sharon Goldwater. 2023. Analyzing acoustic word embeddings from pre-trained self-supervised speech models. In *2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Shane Settle, Keith D. Levin, Herman Kamper, and Karen Livescu. 2017. Query-by-example search with discriminative neural acoustic word embeddings. In *18th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 2874–2878. ISCA.

Shane Settle and Karen Livescu. 2016. Discriminative acoustic word embeddings: Tecurrent neural network-based approaches. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 503–510.

Jan Svec, Jan Lehecka, and Lubos Smídl. 2022. Deep LSTM spoken term detection using wav2vec 2.0 recognizer. In *23rd Annual Conference of the International Speech Communication Association (Interspeech)*, pages 1886–1890. ISCA.

Drisya Vasudev, Suryakanth V. Vasudev, K. K. Anish Babu, and K. S. Riyas. 2016. Combined mfcc-fbcc features for unsupervised query-by-example spoken term detection. In *Intelligent Systems Technologies and Applications*, pages 511–519. Springer International Publishing.

Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R. Scott. 2019. Multi-similarity loss with general pair weighting for deep metric learning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5017–5025.

Yu-Hsuan Wang, Hung-Yi Lee, and Lin-Shan Lee. 2018. Segmental audio word2vec: Representing utterances as sequences of vectors with applications in spoken term detection. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6269–6273.

Yougen Yuan, Cheung-Chi Leung, Lei Xie, Hongjie Chen, Bin Ma, and Haizhou Li. 2018. Learning acoustic word embeddings with temporal context for query-by-example speech search. In *19th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 97–101. ISCA.

Kun Zhang, Zhiyong Wu, Jia Jia, Helen M. Meng, and Binheng Song. 2019. Query-by-example spoken term detection using attentive pooling networks. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 1267–1272. IEEE.

Turkish Native Language Identification

Ahmet Yavuz Uluslu
Universität Zürich & PRODAFT
ahmetyavuz.uluslu@uzh.ch

Gerold Schneider
Universität Zürich
gschneid@cl.uzh.ch

Abstract

In this paper, we present the first application of Native Language Identification (NLI) for the Turkish language. NLI involves predicting the author's native language by analysing their writing in other languages. While most NLI research has focused on English, our study extends its scope to Turkish. We used the recently constructed Turkish Learner Corpus and employed a combination of three syntactic features (CFG production rules, part-of-speech n-grams and function words) with L2 texts to demonstrate their effectiveness in this task.

1 Introduction

Native Language Identification (NLI) is the task of automatically identifying the native language (L1) of an individual based on their linguistic productions in another language (L2). The underlying hypothesis is that the L1 influences learners' second language writing as a result of the language transfer effect (Yu and Odlin, 2016). It is used for a variety of purposes including forensics applications in cybercrime (Perkins, 2018) and secondary language acquisition (Swanson and Charniak, 2014).

Research in NLI is mainly conducted with learner corpora, which comprise collections of writings by individuals learning a new language. These writings are annotated with metadata such as the author's native language (L1) or their fluency level. Recent NLI studies on languages other than English include Portuguese (del Río Gayo et al., 2018), Arabic (Malmasi and Dras, 2014a), and Chinese (Malmasi and Dras, 2014b). The learner corpus is the backbone of NLI research, which means that extending research to a novel language depends on acquiring the appropriate learner corpora for that language. In the past, studies have focused on L2 English because of the prominence of this language in language research and the relatively large amount of data available. To the best of our knowledge, this study presents the first detailed NLI experiments on L2 Turkish. We employ the recently constructed Turkish Learner Corpus (TLC) (Anna, 2022) and investigate widely used linguistic features for NLI. The remainder of the paper is organised as follows: Section 2 discusses related work in NLI, Section 3 and 4 describes the methodology and dataset used in our experiments, and Section 5 presents the experimental results. Finally, Section 6 presents a brief discussion and concludes this paper with directions for further research.

2 Related Work

NLI is typically modeled as a supervised multi-class classification task. In this experimental design, the individual writings of learners are used to train a model while the author's L1 information serves as class labels. A variety of feature types at the syntactic and lexical levels were studied to capture distinct characteristics of the language interference phenomenon: spelling errors, word and lemma n-grams, dependency parsing, and morphosyntax. A more detailed review of feature extraction-based methods can be found in two shared task reports on the NLI task organised in 2013 and 2017 (Tetreault et al., 2013; Malmasi et al., 2017).

In recent years, there has been increased experimentation with deep learning methods, including pre-trained transformers (Steinbakken and Gambäck, 2020) and generative models (**?**). While these models slightly outperformed the state-of-the-art performance achieved by feature-based stacked classifiers, questions about their interpretability, inherent biases, and practical shortcomings in industrial applications remain unexplored. Traditional methods based on hand-crafted features continue to be preferred in many implementations due to their simplicity in training and resource efficiency. Within this context, Uluslu and Schneider (2022) approached the NLI scalability problem in the context of cybercrime through the use of adapter fine-tuning.

3 Data

In this study, we use data from the TLC (Anna, 2022). TLC is a learner corpus composed of the writings of learners of Turkish. These texts are essays written as part of a test of Turkish as a secondary language. Each text includes additional metadata such as the nationality of the author and the genre of the text. The corpus also includes error codes and corrections, although we do not make use of this information.

We used a subset of the dataset containing texts for five L1 groups: Arabic (ARA), Albanian (AL), Azeri Turkish (AZ), Farsi (IR), and Afghani (Pashto) (AFG). We chose these five languages mainly because of the immigration trend observed in Turkey, which will result in a need for additional capabilities for cybercrime forensics and language learning applications for educational institutions. We limit our study to the genre of essays. The other genres (letters of different forms) in the corpus are unbalanced and scarce which may introduce linguistic biases across different registers. We also do not attempt to adjust the dataset based on the writing prompts because they were unbalanced across languages. We randomly selected sentences from the same L1 and combined them to produce documents of approximately the same length. This methodology ensures that the texts for each L1 are a mix of different authoring styles, topics, and language proficiency. The composition of our data is shown in Table 1.

L1	Docs	Tokens	TTR	Avg Words
AFG	55	12546	0.47	278.8
AL	58	15001	0.52	250.6
ARA	65	16969	0.49	252.9
AZ	54	15850	0.47	273.5
IR	52	12870	0.51	246.4

Table 1: Distribution of the five L1s in terms of texts, tokens, type/token ratio (TTR) and average words.

4 Methodology

4.1 Classifier

In our study, we use the standard supervised multi-class classification approach for NLI. A linear Support Vector Machine (SVM) is used for classification and feature vectors are created using a TF-IDF weighting scheme, in line with previous research (Gebre et al., 2013). We initially experimented with relative frequencies but obtained better preliminary results with TF-IDF. We performed a grid search in parameter space for the regularisation parameter C in the range 10e-6 to 10e-1 and set max_iteration = 5k to ensure model convergence. We find that the generalisation of the model reaches its limit at C = 1, we, therefore, choose this value.

4.2 Evaluation

Following the previous NLI studies, we present our findings using classification accuracy through 10-fold cross-validation (10FCV), which has become the standard for NLI result reporting in recent years. Our cross-validation approach is randomised and stratified, aiming to maintain consistent class proportions across partitions. Since our dataset is slightly unbalanced, we provide detailed metrics in addition to accuracy, including precision per class, recall, and F1 values. We also compare these results to a random baseline.

4.3 Linguistic Features

We focus only on content-independent features, in particular syntactic features, following the example of studies on NLI in other languages (Malmasi et al., 2015). Due to the imbalance in the topic distribution in the TLC corpus, we decided not to include lexical features such as word n-grams and embeddings in our study. Topic bias can arise when certain subjects or topics are not equally represented across different classes (Brooke and Hirst, 2013). For example, only students with Azeri and Farsi L1 were asked to respond to prompts specifically about happiness and time. This can result in the classifier to associate these topics with the languages, rather than discerning the linguistic characteristics inherent to Azeri and Farsi, thereby introducing a confounding variable to the task. Even if we attempted to balance the topics across languages, similar to the TOEFL11 dataset (Blanchard et al., 2013), we found that particular rhetoric strongly influences certain backgrounds. For example, writings of the students from Afghanistan were predominantly religious, regardless of the topic. By focussing on syntactic features, we aim to capture the underlying syntactic influence of the L1 on its L2 writing independently of the content. We experimented using a combination of three syntactic features: context-free grammar (CFG) production rules, part-of-speech n-grams, and function words.

Function words are content-independent words, including prepositions, articles, and auxiliary verbs, that play a crucial role in conveying grammatical relationships between words. It is often challenging for L2 speakers to use the appropriate function words and production errors may be due to the influence of their L1 (Schneider and Gilquin, 2016). These function words are recognised as valuable features for the NLI task. We extracted frequencies of 75 Turkish words from different grammatical categories. However, it's worth noting that many grammatical aspects, which are morphologically expressed in Turkish, may not be as strongly captured as they would be in English.

Part-of-Speech tags are linguistic categories or word classes that signify the syntactic role of each word in a sentence. They include basic categories such as verbs, nouns, and adjectives. Assigning POS tags to words in a text introduces a level of linguistic abstraction, meaning that we can work with the underlying structure rather than the content. We use the Turkish POS module from Stanza (Qi et al., 2020) to extract universal POS tags, from which we create n-grams of sizes 1 to 3. These n-grams serve to capture preferences for specific word classes and their localised ordering patterns. Our experiments indicated that sequences of order 4 or higher lead to lower accuracy due to the limited size of our corpus. Therefore, we excluded such higher-order n-grams from our analysis.

Hızlı	*kahverengi*	*tilki*	*ve*
ADJ	ADJ	NOUN	CONJ
tembel	*köpeğin*	*üzerinden*	*atlar.*
ADJ	NOUN	POSTP	VERB

3-gram Example: (NOUN, POSTP, VERB)
Functional n-gram Example: (ve, üzerinden)
CFG Rule example: (NP → ADJ NOUN)

Figure 1: An example of a Turkish sentence and feature extractions for POS n-grams, function word n-grams and CFG-Rule extractions.

CFG production rules are used to generate constituent parts of sentences, such as noun and verb phrases. We use the Turkish parsing module of Stanza (Qi et al., 2020) to extract the constituency tree for the documents. The production rules are then extracted and each rule is used as a standalone feature. We exclude lexicalizations to focus on more abstract and general syntactic patterns. These production rules can encode highly idiosyncratic constructions that are specific to particular L1 groups. They have been widely utilized in various ensemble methods for NLI and have been shown to complement other features effectively (Malmasi and Dras, 2018).

5 Results

In this section, we present the results in terms of accuracy achieved by individual feature types. Subsequently, we report the performance obtained using the combination of all features. Finally, we examine the performance obtained by the best system for each L1 class.

Feature Type	Accuracy (%)
Random Baseline	20.0
POS 1-grams	33.4
POS 2-grams	38.9
POS 3-grams	38.6
Function Words	37.2
CFG Production Rules	41.4
Full Combination	**44.2**

Table 2: 10-FCV Accuracy Classification Results

Table 2 displays the results of the systems trained with different feature types in terms of accuracy. We found that all feature types individually outperform the baseline. The CFG rules are the features that individually perform the best, achieving an accuracy of 41.4%. This demonstrates the importance of the syntactic differences between the L1 groups. The full combination, using all feature types, obtains performance higher than CFG features achieving 44.2% accuracy. These trends are very similar to previous research using the same features (del Río Gayo et al., 2018; Malmasi and Dras, 2014a) with comparable corpora.

L1	Precision	Recall	F1-score
AFG	0.50	0.29	0.37
AL	0.45	0.54	0.49
ARA	0.43	0.65	0.52
AZ	0.47	0.41	0.44
IR	0.37	0.15	0.21
Average	0.44	0.41	0.40

Table 3: Full combination per-class results: precision, recall and the F1-score.

Table 3 shows the results obtained for each L1 in terms of precision, recall, and F1 score, as well as the average results for the five classes. Across all classes, we obtain a micro-averaged F1 score of 0.40 and a macro-averaged F1 score of 0.44.

To provide a visual representation of these findings and to highlight any error patterns, we present a heatmap confusion matrix of the classification errors in Figure 2.

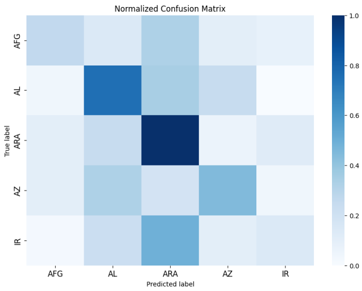


Figure 2: Confusion Matrix of Classification Errors

In most NLI studies, the expected difficulty is to distinguish closely related L1s that may belong to the same language family. Based on the analysis of the confusion matrix, the most notable confusion occurs between Persian and Arabic, both of which have strong lexical connections to the Turkish language. However, since we are working with content-independent features, we attribute this confusion to corpus representation and do not seek any linguistic explanations. We observed no confusion between Afghani and Persian, even though both languages belong to the same language family and have strong similarities. A previous study in comparable settings also failed to offer strong interpretations based on sociolinguistic insights in the error analysis (Malmasi and Dras, 2014a).

Our analysis brings attention to two potential limitations that might prevent drawing connections between model errors. Firstly, although the size of our corpora is relatively limited when compared to other NLI studies—being five times smaller than the Portuguese corpus reported by del Río Gayo et al. (2018) and ten times smaller than the Norwegian corpus described in Malmasi et al. (2015)—it is comparable to the corpus size used for Arabic (Malmasi and Dras, 2014a), which achieved a similar performance compared to our study. The difference in data size and quality might explain the model's generalisation capabilities. Secondly, we acknowledge that our parser might not be entirely suitable for learner language, which could introduce additional noise into the feature space (Van Rooy and Schäfer, 2009).

6 Conclusion & Discussion

In this study, we presented the first experiments with Turkish NLI and achieved a level of performance comparable to previous results for other languages. Our main focus was to investigate the effectiveness of syntactic features for Turkish, a language that differs from English in certain aspects, particularly in morphological complexity. Another significant contribution of our work is the introduction of a new dataset for NLI, specifically designed to address L1-based language transfer effects. This corpus can serve as a valuable resource for researchers to validate and refine their methodologies across various datasets and languages.

We identify several promising directions for future research. Firstly, we plan to expand the corpus by incorporating more learner writings and extending the analysis to encompass other L1 languages. Additionally, we believe that assessing the proficiency level of learners can shed further light on the observed challenges. Finally, we plan to explore more linguistically sophisticated features in our investigation. For instance, leveraging L1 mistakes from a morphological perspective as a content-independent feature could yield valuable insights. To this end, our follow-up study will incorporate a broader range of features to enhance the robustness and comprehensiveness of our analysis.

Ethics Statement

Our study only processes information from publicly available learner corpora. We place great emphasis on protecting the privacy of individuals and ensure that no sensitive personal data is accessed, stored or processed at any stage of the project. Our research adheres to the ethical guidelines of the University of Zurich and PRODAFT.

Acknowledgements

This work was partially supported by the collaboration between the University of Zurich and PRODAFT within the Innosuisse Innovation Project AUCH (103.188 IP-ICT).

References

Golynskaia Anna. 2022. An error coding system for the turkish learner corpus. *The Journal of Linguistics*, (39):67–87.

Daniel Blanchard, Joel Tetreault, Derrick Higgins, Aoife Cahill, and Martin Chodorow. 2013. Toefl11: A corpus of non-native english. *ETS Research Report Series*, 2013(2):i–15.

Julian Brooke and Graeme Hirst. 2013. Native language detection with 'cheap'learner corpora. In *Twenty Years of Learner Corpus Research. Looking Back, Moving Ahead: Proceedings of the First Learner Corpus Research Conference (LCR 2011)*, volume 1, page 37.

Iria del Río Gayo, Marcos Zampieri, and Shervin Malmasi. 2018. A Portuguese native language identification dataset. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 291–296, New Orleans, Louisiana. Association for Computational Linguistics.

Binyam Gebrekidan Gebre, Marcos Zampieri, Peter Wittenburg, and Tom Heskes. 2013. Improving native language identification with TF-IDF weighting. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 216–223, Atlanta, Georgia. Association for Computational Linguistics.

Shervin Malmasi and Mark Dras. 2014a. Arabic native language identification. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 180–186, Doha, Qatar. Association for Computational Linguistics.

Shervin Malmasi and Mark Dras. 2014b. Chinese native language identification. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 95–99, Gothenburg, Sweden. Association for Computational Linguistics.

Shervin Malmasi and Mark Dras. 2018. Native language identification with classifier stacking and ensembles. *Computational Linguistics*, 44(3):403–446.

Shervin Malmasi, Mark Dras, and Irina Temnikova. 2015. Norwegian native language identification. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 404–412, Hissar, Bulgaria. INCOMA Ltd. Shoumen, BULGARIA.

Shervin Malmasi, Keelan Evanini, Aoife Cahill, Joel Tetreault, Robert Pugh, Christopher Hamill, Diane Napolitano, and Yao Qian. 2017. A report on the 2017 native language identification shared task. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 62–75, Copenhagen, Denmark. Association for Computational Linguistics.

Ria Perkins. 2018. The application of forensic linguistics in cybercrime investigations. *Policing: A Journal of Policy and Practice*, 15.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Gerold Schneider and Gaëtanelle Gilquin. 2016. Detecting innovations in a parsed corpus of learner english. *International Journal of Learner Corpus Research*, 2:177–204.

Stian Steinbakken and Björn Gambäck. 2020. Native-language identification with attention. In *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 261–271, Indian Institute of Technology Patna, Patna, India. NLP Association of India (NLPAI).

Ben Swanson and Eugene Charniak. 2014. Data driven language transfer hypotheses. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 169–173, Gothenburg, Sweden. Association for Computational Linguistics.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A report on the first native language identification shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 48–57, Atlanta, Georgia. Association for Computational Linguistics.

Ahmet Yavuz Uluslu and Gerold Schneider. 2022. Scaling native language identification with transformer adapters. In *Proceedings of the 5th International Conference on Natural Language and Speech Processing (ICNLSP 2022)*, pages 298–302, Trento, Italy. Association for Computational Linguistics.

Bertus Van Rooy and Lande Schäfer. 2009. The effect of learner errors on pos tag errors during automatic pos tagging. *Southern African Linguistics and Applied Language Studies*, 20:325–335.

Liming Yu and Terence Odlin. 2016. *New perspectives on transfer in second language learning*, volume 92. Multilingual Matters.

KMD: A New Kurdish Multilabel Emotional Dataset For the Kurdish Sorani Dialect

Soran Badawi
Charmo Center for Research, Training and Consultancy, Charmo University, KRG, Iraq
soran.sedeeq@charmouniversity.org

Abstract

This paper presents a new Kurdish Multilabel Emotional Dataset (KMD) for the Kurdish Sorani dialect, which contains emotional labels for four different categories: Fear, Sadness, Joy, and Surprise. The dataset was collected using Twitter API, pre-processed, and manually labelled by three independent annotators. We also conducted experiments using classical machine learning classifiers, including Naive Bayes and Support Vector Machine (SVM), and deep learning models, BLSTM and BERT, to evaluate the efficiency of the dataset. Our results show that the multilingual BERT model outperforms the traditional machine learning classifiers in emotional labelling accuracy. The KMD dataset can be used for various natural language processing tasks, including sentiment analysis, emotion detection, and opinion mining, for the Kurdish Sorani dialect.

1 Introduction

Emotions play a significant role in human communication. The presence and significance of emotions can be found in every area of our existence. Our emotions impact the choices we make, how we interact with others, and our daily conduct (Erol et al., 2019). They even persist beyond our memories. As the quantity of text-based content that conveys emotions grows rapidly (e.g., microblog posts, blogs, and forums), there is a pressing demand and potential to create automated tools that can recognize and evaluate emotions expressed in written language. In many situations, machines must engage with or observe humans, and emotion recognition has numerous practical uses in such settings. For example, in an online learning platform, an automated tutor could give more effective feedback to a student by taking into account her level of motivation or frustration. Similarly, a car that has the capability to assist a driver could take action or sound an alert if it detects that the driver is fatigued or anxious (Kosti et al., 2019).

Therefore, recognizing emotions is an essential task for machines to understand human behaviour. Natural Language Processing (NLP) is a rapidly growing field that aims to enable machines to understand human language (Khurana et al., 2023). One of the essential components of NLP is the availability of labelled datasets that can be used for the training and evaluation of machine learning models. However, there is a lack of such labelled datasets for some languages, including the Kurdish Sorani dialect.

In this paper, We endeavour to construct an emotion detection system for the Kurdish language, utilizing the Sorani dialect. Our efforts involved creating the initial emotion-annotated dataset for the Kurdish language entirely annotated by human annotators. Additionally, we utilized both machine learning classifiers and deep learning models to train our dataset and documented the findings of our experiment. We also conducted an experiment on the KMD dataset using classical machine learning classifiers and deep learning BLSTM and BERT to evaluate the efficiency of the dataset. For the machine learning classifiers, Naive Bayes and Support Vector Machine (SVM) classifiers were implemented. These models were selected because of their wide usage in machine-learning workflows. The availability of the KMD dataset will enable researchers to develop and evaluate emotion recognition models for the Kurdish Sorani dialect. The dataset will also be useful for various NLP applications, such as sentiment analysis and opinion mining. In summary, this paper presents a significant contribution to the field of NLP and emotional analysis by introducing a new dataset for the Kurdish Sorani dialect.

The following sections comprise the remainder of this paper: Section 2 provides the related works, while Section 3 introduces the data collection methods. In Section 4, we detail our robust baselines and experiments. Sections 5 and 6 discuss the findings. Lastly, we conclude the paper in Section 7.

2 Related Works

The Kurdish language belongs to the Indo-Iranian branch of the larger Indo-European language family and consists of 33 letters. It is relatively similar to Persian and is spoken by approximately 30-40 million people in Iran, Turkey, Iraq, and Syria (Badawi, 2023b). In Iraq, the Kurdish language is recognized as one of the official languages (Ahmadi et al., 2019). There are two main dialects of Kurdish: Central Kurdish (Sorani) and Northern Kurdish (Badawi, 2023c). However, several minor dialects, such as Gorani (Hawrami), are used by small communities in Iraq and Iran, and Zazaki spoken in Turkey (Buran, 2011).

There have been several efforts to create emotional datasets for different languages, such as English (Plaza del Arco et al., 2020), Chinese (Feng et al., 2022), Arabic (Al-Khatib and El-Beltagy, 2017), and Hindi (Singh et al., 2022). However, the efforts on the Kurdish language increasingly focused on building sentiment analysis datasets. Sentiment analysis is a subfield of NLP that aims to identify and extract opinions and emotions expressed in text. Several studies have focused on sentiment analysis for Kurdish Sorani. Awla and Veisi (Awlla and Veisi, 2022) built a sentiment analysis dataset from gathering Facebook comments. A total of 18,450 comments were extracted from 13 popular pages. After collecting the data, the authors performed preprocessing on the comments by removing noisy comments and those that were not written in the Central Kurdish language. Three annotators were assigned to annotate each comment: Positive, Negative, or Neutral. The work of Hameed, Ahmedi and Rezai (Hameed et al., 2023)is another example of dataset construction in the field of sentiment analysis. Using Twitter API, the authors were able to collect and annotate 1769 tweets. The labels include positive, negative, mixed, neutral, and none. Furthermore, several studies worked on building Kurdish datasets in the field of text classification. Currently, we are aware of only two annotated corpora. The first one is the medical corpus, which contains 6756 samples obtained from Facebook comments and is divided into medical and non-medical (Saeed et al., 2022). The second one is KDC-4007, comprising 4,007 text files categorized into eight groups: Sports, Religions, Arts, Economics, Education, Socials, Styles, and Health (Rashid et al., 2018). Notably, no datasets for detecting emotions in the Kurdish language are currently available. Finally, KNDH (Kurdish News Dataset Headlines) is a dataset which includes a collection of 50,000 news headlines, equally distributed among Health, Science, Social, Economic and Sports categories (Badawi et al., 2023).

There is a concerning scarcity of annotated datasets for emotion detection in the Kurdish language. Previous studies that have focused on dataset creation for the Kurdish language have primarily focused on sentiment analysis, which involves categorizing texts into positive, negative, or neutral categories based on the expressed sentiment. However, emotions are more complex than simply positive or negative sentiments. Our work aims to fill this gap by building a new emotional dataset for the Kurdish Sorani dialect that includes a wider range of emotions, such as fear, sadness, joy, and surprise. By including a broader range of emotions, we can gain a deeper understanding of the nuances of language and how emotions are expressed in different contexts. This will enable us to build more accurate and effective natural language processing models that can be used for a variety of applications, including sentiment analysis, emotion detection, and text classification.

3 Dataset Benchmark

3.1 Data collection

To gather data, either a software program or specialized libraries must be employed through coding. In this study, we utilized the latter method and leveraged the Twitter developer API to extract tweets while removing user identities to comply with Twitter's policies and ensure security. The dataset resulting from this process is freely available on the Mendeley repository, accessible via the*URL ,[1]. Figure 1 outlines the steps taken to construct this dataset.

Undoubtedly, raw data can be contaminated with noise. Online Kurdish data, for instance, often contains words from other languages, special characters, elongated letters, symbols, and irrelevant numbers. During the preprocessing phase, we removed all non-Kurdish characters from HTML links. In the second phase, special characters were examined. If a specific character is used to convey senti-

[1] https://data.mendeley.com/datasets/dntxt73dm6/1

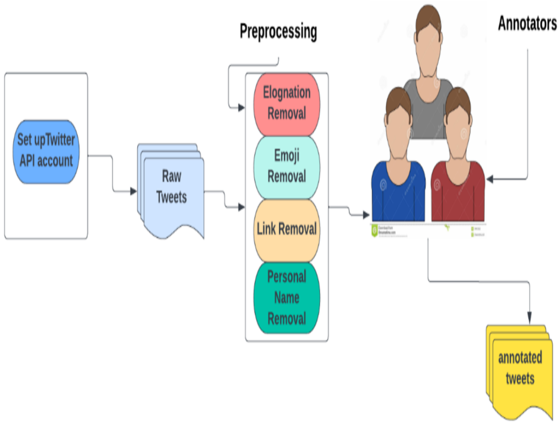


Figure 1: Data collection process

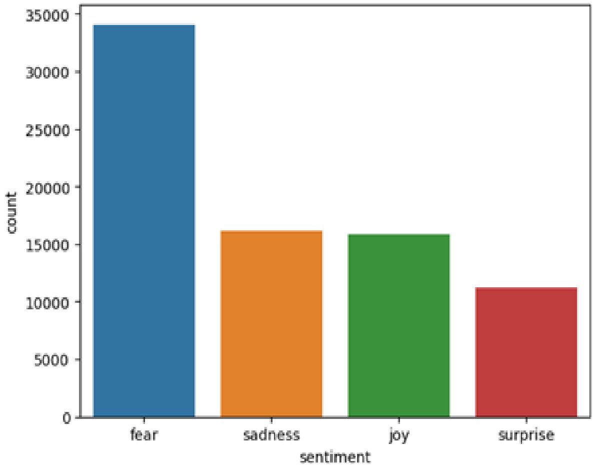


Figure 2: Class distribution in the dataset

Annotated	R1 + R2	R1 + R3	R2 + R3
kappa	0.88	0.79	0.86

Table 1: KAPPA score among raters, R1,R2,R3 refer to Rater 1, Rater 2 and Rater 3 respectively

ment, it is left unchanged. Characters that have no meaning were removed from the text. Sometimes, numbers can express feelings or sentiments for the user, and therefore, they are included in the text.

3.2 Annotation Process

A group of three individuals who are knowledgeable in the Kurdish language were selected to annotate the corpus. These annotators have an educational background in the Kurdish language. To assist them in the annotation process, the annotators were provided with instructions, which included the following:

1. The annotators were required to determine the label of each texts.

2. The corpus was updated with the most noteworthy annotations for each piece of text, based on the number of votes it received.

The entire dataset was given to the annotators. The degree of agreement between the annotators was measured using Kappa coefficients (Cao et al., 2016). The Kappa coefficients demonstrated a strong level of agreement during the sentiment annotation process, ranging from 0.79 to 0.88, as presented in Table 1. These numbers are within an acceptable range according to previous studies.

3.3 Dataset Statistics

The whole dataset consists of 77270 texts distributed among fear, sadness, joy and surprise labels as depicted in Figure 3.

Figure 2 shows that fear was the most commonly existing emotion in the dataset, as it had the highest frequency count of 34085. The second most common emotion was sadness, with a frequency count of 16099. Joy was reported less frequently than both fear and sadness, with a frequency count of 15857. On the other hand, surprise was the least frequently reported emotion with a frequency count of 11204.

4 Experiment

In this paper, we aim to conduct an experiment on the KMD dataset using both classical machine classifiers and deep learning BLSTM and BERT to evaluate the efficiency of the dataset. For the machine learning classifiers, Naive Bayes and Support Vector Machine (SVM) classifiers were implemented. These models were selected because of their wide usage in machine-learning workflows.

The Naive Bayes algorithm is a probabilistic algorithm that is commonly used for text classification tasks (Abbas et al., 2019). Specifically, the Multinomial Naive Bayes variant of the algorithm is often used in natural language processing applications. To implement the Naive Bayes classifier in the paper, the authors used the Scikit-Learn library in Python. The library provides an implementation of the Multinomial Naive Bayes algorithm that is easy to use and has good performance. We first preprocessed the data by filtering out noisy comments and non-Central Kurdish letters. Then, the data was split into training and testing sets with a ratio of 80:20. The training set was used to train the classifier, while the testing set was used to evaluate its performance. The next step was to convert the text data into numerical feature vectors that can be used as input to the classifier. We used a technique called CountVectorizer to represent the text data as feature vectors. CountVectorizer is a commonly used technique in text classification that converts text into a matrix of token counts. Once the data was preprocessed and converted into feature vectors using CountVectorizer, We trained the Naive Bayes classifier on the training set. The trained classifier was then evaluated on the test set to measure its accuracy in predicting the emotional sentiment of the tweets.

SVM (Support Vector Machine) is a popular machine-learning algorithm used for classification tasks. It finds the best hyperplane that separates the data into different classes by maximizing the margin between the hyperplane and the nearest data points (Cervantes et al., 2020). In this paper, We used the SVM algorithm as one of the machine learning classifiers to evaluate the efficiency of the KMD dataset. The Scikit-learn library in Python was used to implement the SVM algorithm.Overall, the SVM classifier was implemented using the Scikit-learn library in Python.

Throughout the experimentation phase, we selected BLSTM and BERT as our preferred options. BLSTM, a neural network type, excels in handling sequential data, encompassing both time series and text data. It comprises of two LSTM layers that process the input sequence in both forward and backward directions, merging their outputs at every time step. This comprehensive approach enables a deeper comprehension of the input sequence's past and future context, enhancing performance for tasks such as sentiment analysis, named entity recognition, and machine translation (Badawi, 2023a)

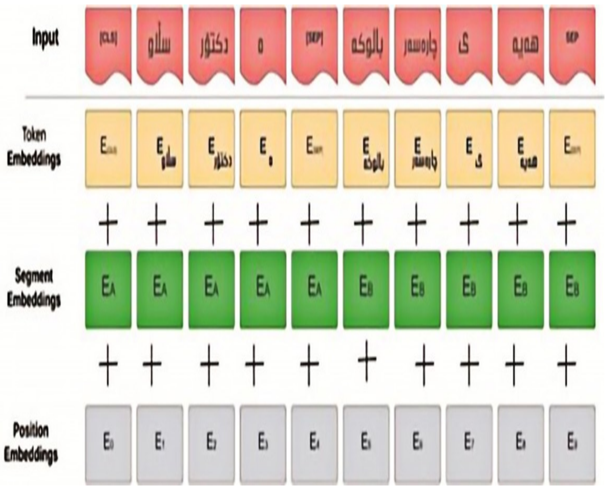


Figure 3: Representation of BERT Iokanization

The BERT model is a deep learning technique that has been widely used for natural language processing tasks (Koroteev, 2021). In this paper, the BERT model is employed for the classification of emotions in the KMD dataset. To implement the BERT model, the training data is tokenized and fine-tuned using the Hugging Face Transformer library. The BERT tokenizer is used to split the text into tokens and add special tokens such as [CLS] and [SEP] as shown in Fig 3. A maximum length of 128 is utilized for the BERT tokenizer. The BERT model is known for its ability to capture the context and meaning of words in a sentence. It achieves this through its attention mechanism, which enables it to focus on important parts of the input text. The fine-tuned BERT model is capable of accurately predicting the emotional sentiment of a given text. It is worth noting that, similar to the machine learning classifiers, the text data used for the BERT model is also represented as feature vectors using CountVectorizer, a text. preprocessing technique that converts the text data into numerical features.

5 Results And Discussions

To establish a basic benchmark for determining the sentiments present in our dataset, we utilized different machine-learning algorithms. The method used for splitting the data can significantly affect the accuracy of the model. Since our dataset is relatively big, we chose to use the holdout method for splitting the data. In the first phase, we used the 80% train 20% test technique. Furthermore, we split the train set using the holdout method to create a validation set. Having a validation set is vital, especially in the case of deep learning. The outcomes obtained from each classifier are presented in Table 2.

The results of the experiments show that all four classifiers (Naïve Bayes, SVM, BLSTM and BERT) have achieved reasonably good performance on the KMD dataset. In general, BERT performed the best among the four classifiers, with an overall F-score of 0.65, followed closely by BLSM with an overall F-score 0.64, SVM with an F-score

Labels	Naive Bayes	SVM	BLSTM	BERT
Fear	0.74	0.75	0.76	**0.77**
Sadness	0.54	0.55	0.56	**0.58**
Joy	0.57	0.56	0.58	**0.58**
Surprise	0.32	0.38	0.42	**0.49**

Table 2: F1-score obtained by each classifiers

Labels	Naive Bayes	SVM	BLSTM	BERT
Fear	0.77	0.78	0.79	**0.80**
Sadness	0.55	0.58	0.59	**0.63**
Joy	0.59	0.57	0.56	**0.60**
Surprise	0.34	0.39	0.43	**0.48**

Table 3: Precision-score obtained by each classifiers

of 0.63, and Naïve Bayes with an F-score of 0.62. The results also show that the classifiers performed differently across the different emotion labels. For the fear and sadness labels, Naïve Bayes and SVM performed similarly with F-scores of around 0.7 and 0.5, respectively, while BLSM and BERT performed slightly better with F-scores of 0.76, 0.56, 0.77 and 0.58, respectively. For the joy label, all four classifiers performed similarly with F-scores ranging from 0.56 to 0.58.

However, for the surprise label, BERT outperformed the other three classifiers with an F-score score of 0.49, while Naïve Bayes and SVM performed much worse with F-scores f 0.32 and 0.38, respectively. However, BLSM performed slightly better with a score of 0.42. This suggests that the surprise label may be more difficult to classify accurately than the other emotion labels in the KMD dataset. The performances of all models are illustrated in Fig 4.

Table 3 displays the precision scores of four classifiers (Naive Bayes, SVM, BLSTM, and BERT) for four different emotions: Fear, Sadness, Joy, and Surprise. Precision is a crucial metric to measure the accuracy of a classifier's positive predictions. The precision scores in the table offer valuable insights into the performance of these classifiers in emotion classification. BERT turns out to be the top-performing classifier in all emotions. It consistently outperforms other classifiers such as BLSTM, SVM, and Naive Bayes. In the "Fear" emotion category, BERT achieves the highest precision score of 0.80, followed closely by BLSTM with a precision of 0.79, while SVM and Naive Bayes lag slightly behind at 0.78 and 0.77, respectively. This trend is observed consistently across the various emotions. BLSTM shows competitive performance, ranking second across all emotions. It closely follows BERT's precision scores, indicating its effectiveness in emotion classification. SVM ranks third in most cases, followed by Naive Bayes, which consistently has the lowest precision scores. This suggests that advanced models like BERT and BLSTM tend to outperform traditional classifiers like SVM and Naive Bayes for this emotion classification task. Notably, the "Surprise" emotion is the most challenging for all classifiers. This is indicated by the especially lower precision scores for "Surprise" compared to the other emotions. The fact that all classifiers struggle to accurately classify "Surprise" suggests that this emotion may have unique characteristics that are difficult to capture, possibly due to its nuanced expression in the dataset or semantic complexities.

Labels	Naive Bayes	SVM	BLSTM	BERT
Fear	0.78	0.79	0.80	**0.83**
Sadness	0.56	0.59	0.61	**0.64**
Joy	0.58	0.56	0.57	**0.61**
Surprise	0.35	0.40	0.44	**0.49**

Table 4: Recall-score obtained by each classifiers

Table 4 provides a comprehensive overview of recall scores for four different classifiers used for emotion classification. Recall, also referred to as sensitivity or the true positive rate, is a critical metric that measures a classifier's ability to correctly identify all positive instances. Upon analyzing the recall scores in the table, we can draw important insights into the performance of these classifiers in the context of emotion classification. A distinct pattern emerges for each of the four emotions. For the "Fear" emotion category, BERT has the highest recall score of 0.83, followed closely by BLSTM at 0.80, SVM at 0.79, and Naive Bayes at 0.78. This pattern continues across the other emotions, where BERT consistently shows the highest recall scores, affirming its position as the best-performing classifier for emotion classification based on recall. The BLSTM model ranks second for most emotions, including "Fear," "Sadness," and "Joy," providing recall scores that are very close to BERT's. SVM and Naive Bayes tend to perform lower in terms of recall performance across all emotions. This consistent trend highlights the superiority of advanced models such as BERT and the effectiveness

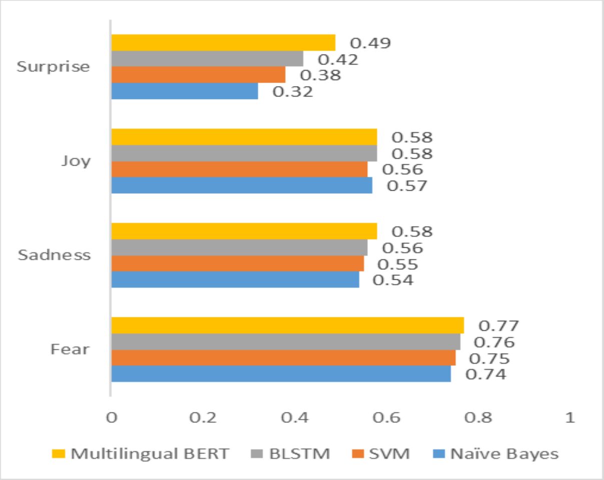


Figure 4: PErformance of each methods

of BLSTM to deliver competitive results in emotion classification tasks. Moreover, it also highlights that the "Surprise" emotion is a considerable challenge for all classifiers, as indicated by their notably lower recall scores for this emotion. This suggests that "Surprise" may possess unique characteristics that make it more difficult to accurately classify. These challenges may stem from the subtle and nuanced expressions associated with this emotion in the dataset or the intricacies of semantic interpretation.

Overall, these results indicate that the KMD dataset can be used to effectively train emotion classifiers for the Kurdish Sorani dialect using both classical machine learning methods and deep learning techniques like BERT. Furthermore, the results highlight the importance of evaluating classifiers across multiple emotion labels to better understand their performance and limitations.

6 Conclusion

The KMD dataset offers a new and valuable resource for researchers interested in the Kurdish Sorani dialect and emotional analysis. The dataset's comprehensive structure, annotation guidelines, and categories provide researchers with a robust foundation for further analysis and experimentation. The proposed model, which utilizes classical machine learning algorithms and deep learning BERT, demonstrates superior performance compared to traditional machine learning algorithms, such as Naive Bayes and Support Vector Machine classifiers and deep learning BLSTM, on all classes. The KMD dataset, along with the proposed model, paves the way for further research on the Kurdish Sorani dialect and provides a benchmark for future studies in natural language processing and machine learning (Badawi, 2023d).

Limitations

The study presents significant findings in the realm of natural language processing and emotional analysis, yet several limitations deserve consideration. Firstly, the study predominantly focuses on the Kurdish Sorani dialect, which may restrict the generalizability of its results to other Kurdish dialects, such as Kurmanji and Gorani. Additionally, due to the extensive dataset used in the study, the training process demanded substantial GPU resources. Furthermore, the Kurdish language shares similarities with the Arabic and Persian alphabets, and users may have employed characters from these languages in their texts. Given the lack of specialized libraries or tools for automatic letter correction or replacement in Kurdish, such texts had to be excluded from the dataset. This removal potentially led to a loss of valuable data and linguistic diversity, particularly considering the low-resource nature of the Kurdish language.

Ethics Statement

This study on the development of a Kurdish Multilabel Emotional Dataset (KMD) for the Kurdish Sorani dialect was conducted with a strong commitment to ethical research practices. The research team recognized the importance of ethical considerations throughout the project, including data collection, annotation, and analysis. This ethical statement outlines the key principles and practices adhered to during the study:

1. **Data Collection and Usage:** Data for the KMD dataset was collected using the Twitter API. We ensured that the data collection process adhered to Twitter's policies and guidelines. All data were obtained without compromising the privacy or consent of Twitter users. No personally identifiable information was collected or disclosed.

2. **Informed Consent:** As the dataset involved publicly available social media content, we did not seek explicit consent from individual Twitter users. However, we ensured that the dataset was used solely for research purposes and that no harm or undue exposure was caused to individuals or communities.

3. **Data Anonymization:** To protect the privacy and anonymity of individuals, all identifying information, including usernames and profile pictures, were removed from the collected data. The dataset was processed to ensure that it contained no personally identifiable information.

Acknowledgements

We would like to extend our heartfelt gratitude to Mr. Fatih Kurt for his invaluable assistance in writing the Python code for the data collection process. His contributions were instrumental in the successful execution of this research project. We would also like to express our sincere thanks to the annotators who dedicated their time and effort to the annotation process. Their meticulous work and commitment were essential in creating a high-quality emotional dataset.

References

Muhammad Abbas, Kamran Ali, Saleem Memon, Abdul Jamali, Saleemullah Memon, and Anees Ahmed. 2019. Multinomial naive bayes classification model for sentiment analysis. *IJCSNS International Journal of Computer Science and Network Security*, 19:62–67.

Sina Ahmadi, Hossein Hassani, and John McCrae. 2019. Towards electronic lexicography for the kurdish language. In *In Proceedings of the sixth biennial conference on electronic lexicograph (eLex)*, pages 881–906, Sintra, Portugal.

Amr Al-Khatib and Samhaa El-Beltagy. 2017. Emotional tone detection in arabic tweets. In *18th International Conference, CICLing 2017*, pages 105–114, Budapest, Hungary. Computational Linguistics and Intelligent Text Processing.

Kozhin Awlla and Hadi Veisi. 2022. Central kurdish sentiment analysis using deep learning. *Journal of University of Anbar for Pure Science*, 16:119–130.

Soran Badawi. 2023a. Data augmentation for sorani kurdish news headline classification using back-translation and deep learning model. *Kurdistan Journal of Applied Research*, 08:27–34.

Soran Badawi. 2023b. Kurdsum: A new benchmark dataset for the kurdish text summarization. *Natural Language Processing Journal*, 05:100043.

Soran Badawi. 2023c. Transformer-based neural network machine translation model for the kurdish sorani dialect. *UHD Journal of Science and Technology*, 7:15–21.

Soran Badawi. 2023d. Using multilingual bidirectional encoder representations from transformers on medical corpus for kurdish text classification. *Aro-The scientific Journal of Koya University*, 11:10–15.

Soran Badawi, Ari Saeed, Sara Ahmed, Peshraw Abdalla, and Diyari Hassan. 2023. Kurdish news dataset headlines (kndh) through multiclass classification. *Data in Brief*, 48:109120.

Ahmet Buran. 2011. Kurds and kurdish language. *Journal of Turkish Studies*, 6:43–57.

Hongyuan Cao, Pranab Sen, Anne Peery, and Evan Dellon. 2016. Assessing agreement with multiple raters on correlated kappa statistics. *Biometrical journal. Biometrische Zeitschrift*, 58:935–943.

Jair Cervantes, Farid García-Lamont, Lisbeth Rodríguez, and Asdrubal Lopez-Chau. 2020. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189–215.

Berat Erol, Abhijit Majumdar, Patrick Benavidez, Paul Rad, Kim-Kwang Raymond Choo, and Mo. Jamshidi. 2019. Toward artificial emotional intelligence for cooperative social human-machine interaction. *IEEE Transactions on Computational Social Systems*, 7:1–13.

Xiang Feng, Keyi Yuan, Xiu Guan, and Longhui Qiu. 2022. An emotion analysis dataset of course comment texts in massive online learning course platforms. *Interactive Learning Environments*, pages 1–15.

Razhan Hameed, Sina Ahmadi, and Fatemeh Daneshfar. 2023. Transfer learning for low-resource sentiment analysis. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 37:Article 111. 14 pages.

Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh. 2023. Natural language processing: State of the art, current trends and challenges. *Multimedia tools and applications*, 82:3713–3744.

Mikhail Koroteev. 2021. Bert: A review of applications in natural language processing and understanding. Computing Research Repository, arXiv:2103.11943. version 2.

Ronak Kosti, Jose Alvarez, Adria Recasens, and Àgata Lapedriza. 2019. Context based emotion recognition using emotic dataset. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:2755–2766.

Flor Miriam Plaza del Arco, Carlo Strapparava, L. Alfonso Urena Lopez, and Maite Martin. 2020. EmoEvent: A multilingual emotion corpus based on different events. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1492–1498, Marseille, France. European Language Resources Association.

Tarik Rashid, Arazo Mustafa, and Ari Saeed. 2018. Automatic kurdish text classification using kdc 4007 dataset. In *The 5th International Conference on Emerging Internetworking, Data and Web Technologies*, pages 187–198.

Ari Saeed, Riam Hussein, Chro Ali, and Tarik Rashid. 2022. Medical dataset classification for kurdish short text over social media. *Data in Brief*, 42:108089.

Gopendra Vikram Singh, Priyanshu Priya, Mauajama Firdaus, Asif Ekbal, and Pushpak Bhattacharyya. 2022. EmoInHindi: A multi-label emotion and intensity annotated dataset in Hindi for emotion recognition in dialogues. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5829–5837, Marseille, France. European Language Resources Association.

iTANONG-DS : A Collection of Benchmark Datasets for Downstream Natural Language Processing Tasks on Select Philippine Languages

Moses Visperas and **Christalline Joie Borjal** and
Aunhel John Adoptante and **Elmer Peramo**
Computer Software Division, Advanced Science and Technology Institute
Department of Science and Technology, Diliman, Quezon City, Philippines
{moses.visperas,christallinejoie.borjal,aunheljohn.adoptante,elmer}@asti.dost.gov.ph

Danielle Shine Abacial
Mindanao State University
- Iligan Institute of Technology
Tibanga, Iligan City, Philippines
danielleshine.abacial@g.msuiit.edu.ph

Ma. Miciella Decano
Far Eastern University - Alabang
Alabang, Muntinlupa City, Philippines
201910804@fealabang.edu.ph

Abstract

Benchmark datasets are crucial for evaluating algorithms and models objectively. They provide a standardized basis for comparisons, promote reproducibility, and drive innovation by establishing baselines and encouraging advancements in the field. Limited benchmark datasets exist for various natural language processing tasks in low-resource languages, including most Philippine languages. As part of iTANONG's 10 billion token dataset initiative, the authors release the first iteration of iTANONG-DS[1], a collection of unlabeled and labeled datasets for different NLP tasks such as sentiment analysis, part-of-speech tagging, named entity recognition for Tagalog, and language modeling for Cebuano.

1 Introduction

Recent years have seen an exponential expansion in the field of natural language processing (NLP), revolutionizing a number of applications including information retrieval, sentiment analysis, and machine translation. Despite these developments, the lack of structured benchmark datasets, particularly for low-resource languages, continues to be a problem in NLP research (Cruz and Cheng, 2020).

While the Philippines present a plethora of languages across all of its islands, Tagalog and Cebuano come out as two of the most prominent and widely-used languages in the countries. Both languages exhibit unique linguistic intricacies that reflect the culture of their respective native speakers. Tagalog is a highly inflected language with a complex system of noun cases, verb conjugations, and prepositions. It also has a rich morphology, with many affixes that can be used to modify nouns, verbs, and adjectives. On the other hand, Cebuano is an agglutinative language, which means that words are formed by adding affixes to a root word. This can make the language seem complex and difficult to learn for speakers of other languages. Cebuano also has a rich system of noun cases, which can be used to indicate the role of a noun in a sentence.

Despite the fact that both languages are widely used by a lot of people, they are still considered to be low-resource languages in the research community. This is mainly because there are not any extensive datasets for them that would be useful for the creation and testing of NLP models and algorithms.

While formal datasets like Wiki-text (Merity et al., 2016) and OSCAR (Ortiz Suárez et al., 2019; Abadji et al., 2022) offer a sizable amount of textual data for Tagalog and Cebuano, they fall short of capturing the subtle nuances of these languages as they are utilized in everyday interactions and social media posts. The ability to comprehend these languages in their natural environments necessitates datasets that faithfully capture the dynamic essence of the language, including its informal expressions, geographical differences, and linguistic patterns found in everyday usage. The current datasets fall short of accurately portraying this heterogeneous landscape, impeding the advancement of NLP research for Philippine languages.

The authors of this work provide a thorough account of their efforts to develop task-built datasets

[1] Datasets are publicly available here: https://huggingface.co/dost-asti

for Tagalog and Cebuano, two widely used Philippine languages, primarily for various NLP applications. Recognizing the necessity for targeted and application-specific datasets, they have meticulously collected resources for sentiment analysis, part-of-speech tagging, named entity identification, and language modeling.

This work intends to encourage and promote NLP research for Philippine languages by offering these task-specific datasets and comprehensive insights into the data collection and processing methods. By enabling academics and practitioners to explore and develop in the context of various languages, these datasets significantly close a gap in the NLP research environment. The authors aspire to facilitate the creation of reliable NLP models and algorithms that can successfully manage the distinct language characteristics and difficulties of Tagalog and Cebuano by making high-quality and contextually rich datasets available.

After presenting an introduction to the existing datasets in Section 2, Section 3 proceeds to outline the dataset curation process employed in the study. The initial two subsections of Section 3 detail the data collection process, data sources, and the preprocessing steps applied to the raw data before creating specialized subsets for various downstream NLP tasks. Subsequent subsections provide a comprehensive explanation of the steps involved in curating distinct labeled datasets. Additionally, they offer a comparative analysis between these newly proposed datasets and the currently available datasets for each individual downstream NLP task, allowing for an evaluation of their quality and suitability. Finally, Section 4 gives an insight to the labeling process done on the data in this study.

2 Related Works

2.1 Monolingual Open-Source Data

Although monolingual data is readily available and widely accessible, there are still limitations with existing datasets such as WikiText-TL (Cruz and Cheng, 2019), NewsPH-NLI (Cruz et al., 2021), and the extensive parallel dataset MT560 (Gowda et al., 2021). While the first two datasets are valuable for creating language models as they are sourced from formal entities, they may not adequately capture the colloquial terms and complex Tagalog expressions used in social media and everyday contexts. On the other hand, the MT560 dataset covers a wide range of Philippine languages but predominantly consists of religious content, which may not be suitable for certain NLP tasks. These limitations underscore the need for more diverse and comprehensive datasets that encompass the intricacies of colloquial language usage and address the specific requirements of various NLP tasks.

2.2 Labeled Task Specific Data

While there are existing task-specific datasets available for certain Philippine languages, such as benchmark datasets for sentiment analysis like the Fake News dataset (Cruz et al., 2020) and the Hate Speech dataset (Neil Vicente Cabasag and Cheng, 2019), the availability of benchmark datasets for other NLP tasks remains limited. In particular, there is a scarcity of benchmark datasets for essential tasks like part-of-speech tagging and named entity recognition (NER). While WikiAnn (Rahimi et al., 2019) offers a considerable NER dataset, its main emphasis is on monolingual Tagalog and may not effectively capture the intricacies of informal language usage where code-switching between languages is prevalent.

3 Methodology

3.1 Data Gathering

To create a comprehensive text corpus, a methodical data collection technique was used, which included a wide range of sources such as government and media websites, social media platforms, and online forums. This multifaceted approach made it possible to collect a wide range of textual information, taking into account different genres, styles, and linguistic nuances.

A variety of scripting tools were used to collect the data effectively, utilizing their many features and functionalities. Notably, the data collection process was automated using tools such Selenium, BeautifulSoup, and snscrape, among others. With the help of these tools, it was possible to browse through many websites, gather pertinent data, and put together a sizable dataset.

Language	Formal	Informal
Tagalog	5,159,917	3,057,180
Cebuano	194,001	1,816,735

Table 1: Total Amount of Lines Gathered Per Language

Following the completion of the data gathering phase, the obtained text data were meticulously

<table>
<tr><th>Token</th><th>Regex Code</th><th>Replacement</th></tr>
<tr><td>emojis</td><td>[\U00010000
\U0010ffff]</td><td>XX_EMOJI</td></tr>
<tr><td>line beaks, feeds, etc.</td><td>([\r\n\t\f\v]+( )*)+</td><td>”.”</td></tr>
<tr><td>URLs that start with http/https</td><td>https?:\/\/([\w\-]+\.)+
([\w \-]+)+(\/[^\s]+)*</td><td>XX_URL</td></tr>
<tr><td><word>@<word>.<word></td><td>[\SÑñ]+@ ([\SÑñ]
+\.)+[\SÑñ]+</td><td>XX_EMAIL</td></tr>
<tr><td>URLs that end with com, net, org, co, us, ph</td><td>([\w\-]+\.)+(com|
net| $org \mid co \mid us \mid ph \mid io$)
(\/[^]+)*</td><td>XX_URL</td></tr>
<tr><td>Starts with @</td><td>@[^\s.,!?]+</td><td>XX_USERNAME</td></tr>
<tr><td>Starts with #</td><td>#[a-zA-ZÑñ0-9_]+</td><td>XX_HASHTAG</td></tr>
</table>

Table 2: Pre-processing done on the corpus, patterned from the work of Velasco et al. (2022)

divided into two major categories: formal and informal texts. The source of the data and its innate qualities served as the foundation for this categorization.

Social media posts from prominent personalities and the government were taken into account as articles for formal writing. These posts retained a formal tone appropriate for official communication channels because they came from reliable sources. Incorporating this subset of social media content served the purpose of capturing the subtleties of formal language usage in a digital setting.

On the other hand, information found through keyword searches of social media data and online forums was included in the category of informal writing. Online communities are renowned for enabling casual dialogue and debate, frequently displaying user-generated content and colloquial language usage. With the help of these sources, it was possible to accurately represent the informality and variety of language idioms that characterize typical online conversations.

By meticulously categorizing the data into formal and informal subsets, it was ensured that the dataset had a vast range of text types ranging from official correspondence to casual internet interactions. Table 1 shows the amount of lines of text gathered per language.

3.2 Pre-processing

A preprocessing methodology inspired by the work of Velasco et al. (2022) was used in this study. By addressing particular components that frequently appear in online textual information, this strategy intended to improve the quality and consistency of the text data. In the preparation phases, special tokens were used to substitute emojis, emails, URLs, and usernames. Sentences containing tokens that had less than three tokens were also removed. These special tokens are highlighted in Table 2.

Additionally, a filtering step was added to the preprocessing pipeline to get rid of phrases that had sentences with less than three tokens. The goal of this phase was to remove very short sentences from the dataset because they are likely to have little semantic relevance and might possibly contribute noise. This criterion was enforced to guarantee that the final dataset had a greater level of coherence and significance.

The goal was to improve the text data's quality, consistency, and interpretability by using this methodical and thorough preprocessing methodology. This would then allow for more reliable and accurate downstream NLP analyses and models.

3.3 Sentiment Analysis

The authors utilized Latent Dirichlet Allocation (LDA) (Blei et al., 2003) – a topic modeling technique – in constructing the sentiment analysis dataset. LDA helped identify relevant topics within the data, ensuring diversity. We randomly selected 9,000 sentences from three LDA-identified subjects.

To enhance dataset quality and granularity, the `GPT 3.5` model was used to classify 500 phrases as positive, negative, or neutral. Additionally, sentence embeddings were obtained with the help of sentence transformers in order to capture semantic nuances.

The extracted embeddings were used as a feature in a label propagation method – leveraging Scikit-learn and a radial basis function kernel – to propagate labeled sentiments to unlabeled data. This process generated a substantial collection of predicted sentiment labels which were then manually validated.

The entire labeling process resulted in a dataset, which contained two subsets: one with two unbalanced sentiment distributions and the other with a balanced distribution. Details of the labeled datasets are shown in Table 3.

Dataset	Positive	Negative	Neutral
Balanced	3000	3000	3000
Unbalanced-A	1203	2827	4970
Unbalanced-B	1354	2825	4821

Table 3: Sentiment Analysis Dataset Size

We benchmarked the datasets for sequence classification using an uncased Tagalog RoBERTa model fine-tuned over 45 epochs using the transformers library provide by Huggingface. In Table 4, we present the validation and test scores, and we also include results for a binary classification task using the HateSpeech dataset (Cruz and Cheng, 2022) for comparison. Despite the differences between our new dataset and the hate speech dataset in terms of content, number of labels, and the presence of code-switching, an interesting observation emerges. Even when we extend the training epochs for our dataset, it consistently yields lower scores compared to training with the hate speech dataset, which reaches early stopping at 15 epochs. This discrepancy in training outcomes underscores the unique challenges and nuances, particularly the code-switching aspect, inherent in our benchmark dataset.

Dataset	Validation Acc.	Test Acc.
Balanced	63.55%	65.33%
Unbalanced-A	75.44%	76.11%
Unbalanced-B	67.5%	68.78%
Hatespeech*	78.07%	99.03%

Table 4: Benchmark scores for sentiment classification using a fine-tuned RoBERTa Model

3.4 Part-of-Speech Tagging

In crafting the POS(Part-of-Speech) tagger dataset, the researchers selected initial data points from the pool of scraped news articles. Sentence tokenization follows this process which involves splitting the articles to individual sentences.

To efficiently handle the annotation process with limited time and minimal human effort, the researchers adapted by using a model called SMT-POST (Nocon and Borra, 2016) - a statistical machine translation approach for POS tagging on the collection of sentences. This model was selected as it achieved a higher score at 84.7% compared to earlier POS token classifiers. In order to select high-quality data points, the researchers applied a few filtering mechanisms such as the removal of five-word sequences or fewer. Overall, there were 3,919 tagged sequences with the MGNN tagset. Table 5 shows the word statistics of the tagged chosen data points.

Category	Count
Number of Words	71,444
Vocabulary Size	15,636
Min words per sequence	6
Max words per sequence	26

Table 5: Word Statistics of Part-of-Speech Text Data

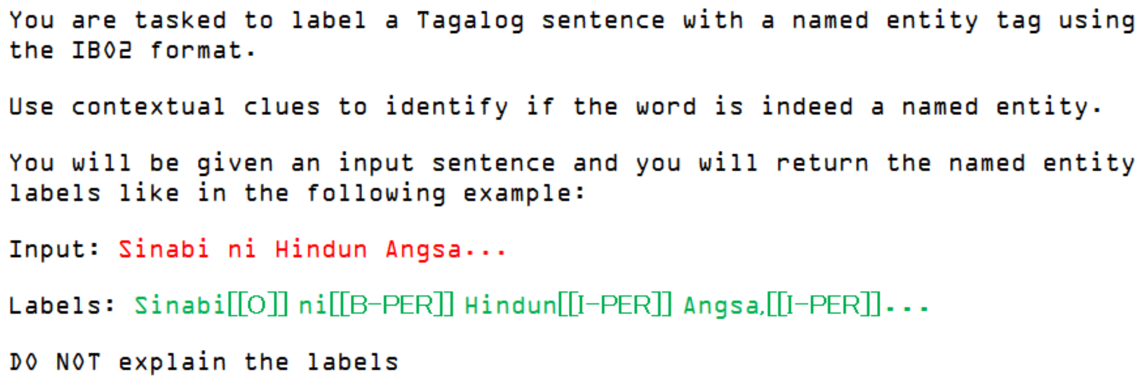
The researchers also observed the POS tags' co-occurrence patterns to demonstrate diversity in terms of syntactical structure. The dataset exhibited a total count of 3,868 unique POS patterns, revealing a wide range of nuances in the collected sentences. To get a clearer observation, they extracted the trigrams in each sequence and calculated the frequency distribution of the corresponding POS tags.

Pattern	Count
FW FW FW	2371
NNP NNP NNP	993
NNC CCB NNC	473
DTP NNP NNP	429
CCT NNC CCB	359

Table 6: Frequency Distribution of Top 5 POS Co-occurrence Pattern

Table 6 shows the top 5 most common POS co-occurrence indicating prevalent code-switching as evidenced by the *FW* tag for foreign words.

The researchers fine-tuned a Tagalog RoBERTa model as a token classifier using the curated POS dataset for 30 epochs. Table 7 displays the validation and test accuracy that achieved a high score of



```
You are tasked to label a Tagalog sentence with a named entity tag using
the IBO2 format.

Use contextual clues to identify if the word is indeed a named entity.

You will be given an input sentence and you will return the named entity
labels like in the following example:

Input: Sinabi ni Hindun Angsa...

Labels: Sinabi[[O]] ni[[B-PER]] Hindun[[I-PER]] Angsa,[[I-PER]]...

DO NOT explain the labels
```

Figure 1: Prompt used for Named-Entity Recognition Task

Dataset	Validation Acc.	Test Acc.
iTANONG-POS	93.10%	92.84%

Table 7: Benchmark scores for Part-of-Speech using a fine-tuned RoBERTa Model

more than 90%. This findings indicate that the prior automated labeling process by SMTPOST resulted to tags with utmost syntactic consistency despite the nuances caused by code-switching. Additionally, iTanong used real-world Taglish texts from media compared to existing researches like CRF-POST(Olivo et al., 2020) that utilized manually translated Wikipedia texts.

3.5 Named Entity Recognition

The researchers gathered 6,230 data points and employed the `GPT 3.5` model to label named entity for each word. To ensure consistent tags, IOB2 tagging format was adopted. In this format, the identified named entity tags are prefixed with *B-* where it begins and *I-* for the subsequent words that are part of the entity. A word that is not a named entity is tagged *O*.

The researchers directed the model with the system payload shown in Figure 1.

The model was instructed with two explicit key points. Firstly, was making contextual inferences to recognize that word entity type may vary depending on the context. For instance, the phrases *Sinabi ng Malacañáng* and *Ginanap sa Malacañáng* uses a common word *Malacañáng* differently as an organizational representative body in one case and as a location of the Philippine President's office in the other.

However, formulating a prompt this way causes `GPT 3.5` to reason out after labeling which results in the excessive generation of tokens. In order to address this issue, another statement was added instructing `GPT 3.5` not to explain the labels. This way of prompting helped the researchers circumvent their problem and ensured the desired result from the model.

Classification	Count
O	124,623
B-PER	4,350
I-PER	4,518
I-ORG	2,773
B-ORG	2,171
I-LOC	1,654

Table 8: Frequency Distribution of 7 Named Entity Tags used in iTanong

The researchers ran the model at a low *temperature* (0.1) to attain a realistic and predictable set of labels. A total of 143,012 named entities were identified by the model with 7 unique classifications. Table 8 shows the distribution of the most frequent tags produced by `GPT 3.5`.

The Tagalog RoBERTa model was fine tuned for 30 epochs to investigate whether the iTANONG NER outperforms WikiAnn in the named entity classification task. It's important to note that the respective test sets for each model were employed, with the iTANONG model being evaluated on the iTANONG test set and the WikiAnn model being assessed on the WikiAnn test set. As depicted in Table 9, WikiAnn performed better at the task by a significant amount.

3.6 Pre-Trained Word Embedding Models

Word embeddings have been a game-changing and groundbreaking force in the field of natural language processing (NLP) ever since they were first used, revolutionizing a variety of tasks within the

Dataset	Validation Acc.	Test Acc.
iTANONG-NER	92.63%	91.10%
WikiAnn*	97.25%	97.53%

Table 9: Benchmark scores for Part-of-Speech using a fine-tuned RoBERTa Model

discipline (Si et al., 2019). These embeddings function as effective numerical representations of words, utilizing the power of neural networks to precisely capture the semantic and grammatical subtleties of words while encoding their contextual essence into multi-dimensional vectors (Cheng, 2022). This outstanding capability has resulted in their undeniable superiority in improving performance across a wide range of downstream NLP activities, establishing their position as a remarkably dependable method of word representation (Ravindran and Murthy, 2021).

In this section, the proponents are pleased to present a set of word embeddings that were developed using the Formal text dataset from the renowned corpus which has been thoroughly detailed in previous sections. These unique embeddings were created utilizing two well-known and distinct techniques, namely Word2Vec (Mikolov et al., 2013) and FastText (Bojanowski et al., 2016). The proponents systematically constructed both the continuous bag of words (CBOW) and skip-gram variants for every technique. Furthermore, each model is provided in a variety of vector dimensions, ranging from the simple 20 to the intricate 300, allowing for a wide range of representation options to choose from. Specifically, there are six different vector sizes to choose from, namely 20, 30, 50, 100, 200, and 300.

The generated models have been saved in both `.bin` and `.txt` formats to enable maximum ease and accessibility while supporting a variety of application scenarios. The combined result of these efforts is an astonishing ensemble of 24 unique models, each of which is available in two different file formats.

3.7 Unlabeled Data & Pre-Trained Language Models

The authors conducted the pretraining of multiple BERT-based language models on the remaining unlabeled data in both Tagalog and Cebuano. Recognizing the necessity of training models from scratch to effectively capture the linguistic subtleties and intricacies inherent to these languages, the research team embarked on this endeavor.

They implemented an 80-20 data split, allocating the available dataset between training and validation sets, thereby enabling meticulous model evaluation and ensuring a robust training process. Despite the existence of Tagalog models, the authors opted for a rigorous training approach on a comprehensive dataset that encompassed both formal and informal language usage. This deliberate incorporation of informal language in the training data was aimed at enhancing the model's ability to adeptly address the diverse linguistic variations encountered in real-world contexts.

In a noteworthy development, the authors pretrained a BERT-Cebuano model, which was a ground-breaking feat. This innovative initiative is, to the best of their knowledge, the first-ever attempt to train a BERT-based language model exclusively for Cebuano. The lack of NLP models designed specifically for Cebuano, a language with few resources and little research attention, is addressed by this study's concentration on Cebuano.

The goal of the work was to build reliable language models that could accurately capture the nuances of Tagalog and Cebuano, hence facilitating subsequent downstream NLP tasks. Results of the model training can be seen on Table 10.

Model	Language	Validation Perplexity
BERT	Tagalog	8.3493
	Cebuano	52.3625
RoBERTa	Tagalog	9.4295
	Cebuano	52.3799

Table 10: Released Pre-Trained Language Models

4 Analysis of Labeling Process

In terms of data labeling, particularly in the context of sentiment analysis where a substantial portion of our labeling process was automated, it's worth noting a limitation. On average, approximately 86% of the entire dataset was correctly labeled through automation – from ChatGPT labeling up to the label propagation. However, there is room for improvement in terms of accuracy, especially if there are plans to expand the dataset with additional labels in the future. Finding more accurate labeling methods or refining the existing automation process could enhance the overall quality of the dataset.

A few challenges were also observed in labelling named entities. Firstly, ChatGPT tend to generate explanations crafted like a user feedback even though it was explicitly prompted to avoid additional details. This impediment resulted to some parsing errors. Then, it produced a lot of redundant tags like *B-PER* and *B-PERSON*. In our earlier attempts, the model produced quite a number of peculiar tags like *B-profanity*, *B-COLOR*, *B-RNA* among others. Finally, the researchers decided to limit the annotation for this iteration into 7 tags, comprising of three broad entity classes *person (PER)*, *location (LOC)* and *organization (ORG)*. The researchers are dedicated to refine the dataset in the next iterations of iTanong-DS from manually scrutinizing the labels to effectively utilizing emerging tools for a semi-automated annotation process.

5 Conclusion

In this paper, the authors present iTANONG-DS, an extensive collection of unlabeled and labeled datasets that have been carefully curated to cater to a wide range of Natural Language Processing (NLP) applications. Specifically, these datasets are designed to facilitate tasks such as sentiment analysis, part-of-speech tagging, named entity recognition, and language modeling for Tagalog and Cebuano languages. Alongside the datasets, they have developed pretrained embeddings and language models specifically tailored to these languages, thereby establishing a strong foundation for NLP research and enabling advancements in Philippine language processing.

In conclusion, iTANONG-DS, along with its accompanying pretrained embeddings and language models, serves as a valuable resource for researchers and practitioners working in the field of Philippine language processing. By providing comprehensive datasets, robust machine learning techniques, and specialized models, this work aims to foster advancements in sentiment analysis, part-of-speech tagging, named entity recognition, and language modeling for Tagalog and Cebuano. It is the hope that the availability of iTANONG-DS will stimulate further research and innovation in NLP for Philippine languages, contributing to the development of sophisticated language technologies and applications tailored to the unique linguistic characteristics of these languages.

Limitations

In the dataset presented in this paper, the tags for sentiment analysis are currently limited to three, and for Named Entity Recognition (NER), there are seven tags. However, there is room for expansion in terms of the number of possible labels. For sentiment analysis, this would involve adding more emotional categories beyond the current three, while for NER, it entails introducing more specific labels. Concurrently, the plan is to increase the number of labeled sentences within this dataset to enhance its comprehensiveness and applicability.

Additionally, it's worth noting that although there is a substantial amount of unlabeled Cebuano dataset, curation of task specific datasets was impossible due to the lack of native Cebuano speakers in the team. Also note that the pre-trained language models may lack capability when dealing with long sequences since the majority of the data used to train the models were taken from social media posts where sequence lengths are limited.

In this paper, the authors also released a collection of pre-trained word embeddings. However, these are only in the word2vec and fasttext formats. GloVE embeddings were not included in the collection.

References

Julien Abadji, Pedro Ortiz Suarez, Laurent Romary, and Benoît Sagot. 2022. Towards a cleaner document-oriented multilingual crawled corpus. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4344–4355, Marseille, France. European Language Resources Association.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2016. Enriching word vectors with subword information. *Computing Research Repository*, arXiv:1607.04606.

Jerome Cheng. 2022. Neural network assisted pathology case identification. *Journal of Pathology Informatics*, 13:100008.

Jan Christian Blaise Cruz and Charibeth Cheng. 2019. Evaluating language model finetuning techniques for low-resource languages. *Computing Research Repository*, arXiv:1907.00409.

Jan Christian Blaise Cruz and Charibeth Cheng. 2020. Establishing baselines for text classification in low-resource languages. *Computing Research Repository*, arXiv:2005.02068.

Jan Christian Blaise Cruz and Charibeth Cheng. 2022. Improving large-scale language models and resources for Filipino. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6548–6555, Marseille, France. European Language Resources Association.

Jan Christian Blaise Cruz, Jose Kristian Resabal, James Lin, Dan John Velasco, and Charibeth Cheng. 2021. Exploiting news article structure for automatic corpus generation of entailment datasets. *Computing Research Repository*, arXiv:2010.11574. Version 3.

Jan Christian Blaise Cruz, Julianne Agatha Tan, and Charibeth Cheng. 2020. Localization of fake news detection via multitask transfer learning. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2596–2604, Marseille, France. European Language Resources Association.

Thamme Gowda, Zhao Zhang, Chris Mattmann, and Jonathan May. 2021. Many-to-English machine translation tools, data, and pretrained models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 306–316, Online. Association for Computational Linguistics.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *Computing Research Repository*, arXiv:1609.07843.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Computing Research Repository*, arXiv:1301.3781.

Sean Christian Lim Mark Edward Gonzales Neil Vicente Cabasag, Vicente Raphael Chan and Charibeth Cheng. 2019. Hate speech in philippine election-related tweets: Automatic detection and classification using natural language processing. *Philippine Computing Journal*, XIV(1).

Nicco Nocon and Allan Borra. 2016. SMTPOST using statistical machine translation approach in Filipino part-of-speech tagging. In *Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation: Posters*, pages 391–396, Seoul, South Korea.

John Francis T. Olivo, Prince Julius T. Hari, and Michael B. dela Fuente. 2020. Crfpost: Part-of-speech tagger for filipino texts using conditional random fields. In *Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence*, ACAI '19, page 444–449, New York, NY, USA. Association for Computing Machinery.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164, Florence, Italy. Association for Computational Linguistics.

Renjith Ravindran and Kavi Murthy. 2021. Syntactic coherence in word embedding spaces. *International Journal of Semantic Computing*, 15:263–290.

Yuqi Si, Jingqi Wang, Hua Xu, and Kirk Roberts. 2019. Enhancing clinical concept extraction with contextual embeddings. *Journal of the American Medical Informatics Association*, 26(11):1297–1304.

Dan John Velasco, Axel Alba, Trisha Gail Pelagio, Bryce Anthony Ramirez, Jan Christian Blaise Cruz, and Charibeth Cheng. 2022. Towards automatic construction of filipino wordnet: Word sense induction and synset induction using sentence embeddings. *Computing Research Repository*, arXiv:2204.03251.

Data Augmentation for Text Classification with EASE

A M Muntasir Rahman[1] **Wenpeng Yin**[2] **Guiling "Grace" Wang**[1]
[1]Department of Computer Science, New Jersey Institute of Technology
[2]Department of Computer Science & Engineering, Penn State University
{ar238,gwang}@njit.edu
wenpeng@psu.edu

Abstract

In this work, we present **EASE**, a simple but dependable Data Augmentation (DA) technique for Text Classification (TC) that has four easy steps: **E**xtract Units, **A**cquire Labels, **S**ift and **E**mploy. We extract meaningful units as augmented samples from original data samples and use powerful tools to acquire labels for them before they are sifted and merged. Previous DA techniques, like EDA-Easy DA (Wei and Zou, 2019) and AEDA-An Easier DA (Karimi et al., 2021), excel with sequential, RNN-based models but struggle with BERT (Devlin et al., 2019) and other transformer-based models that heavily rely on token order. EASE, in contrast, performs well with these models, demonstrating stability, speed, and minimal adverse effects. We tested our intuitive method on multiple challenging datasets sensitive to augmentation, and experimental results have indicated the efficacy of DA with EASE.

1 Introduction

DA is a fairly common technique in Machine Learning, especially in Computer Vision and Speech Recognition, and there are many standard ways of doing it. For example, simply flipping or rotating an image and labeling it the same as the original sample is quite logical. While these techniques do involve elements of randomness, they can still be regarded as logically labeled samples, distinct from random noise. This distinction is essential for enhancing the interpretability of complex deep learning models, a challenge often encountered in several notable NLP DA techniques, including EDA (Wei and Zou, 2019) and AEDA (Karimi et al., 2021), among others.

In EDA (Wei and Zou, 2019), four random operations—Random Synonym Replacement, Random Insertion, Random Swap, and Random Deletion—are employed. These operations, when applied even moderately, can significantly alter the original text's meaning in text classification. A single token replacement, for instance, can reverse the sentiment of a sentence. Similarly, In AEDA (Karimi et al., 2021), random punctuation marks like question marks and periods are inserted into samples, radically altering sentence structure and causing confusion in the training model. Despite their proven effectiveness in ideal situations, these techniques often hinder performance. Particularly in the era of Transformers (Vaswani et al., 2017), where positional encodings are crucial and depend on token order, random rearrangement disrupts the models' contextual understanding. Hence, the demand for a DA method that accounts for this critical aspect became apparent.

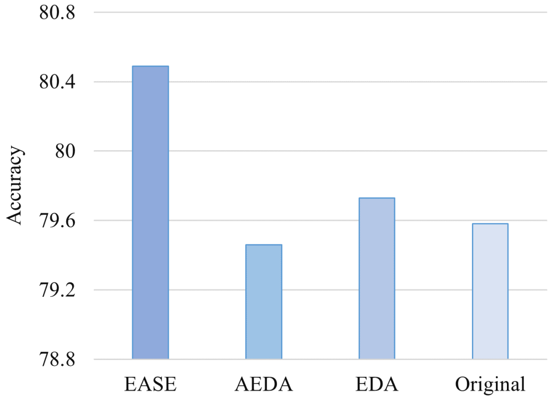


Figure 1: Averaged accuracy accross all datasets and models used in the low-resource experiments.

.

The rise of large language models, such as BERT-base (110M parameters), necessitates a DA (DA) technique that avoids substantial expansion of the training set and the associated increase in training time. Notably, EDA and AEDA suggest a substantial 9-10 times dataset size augmentation, significantly impacting fine-tuning duration. Furthermore, transformer-based models have eclipsed RNN-based models, rendering experiments with EDA or AEDA on the latter obsolete. These models' potent bidirectional contextual representations demand robust DA methods and more challenging datasets. Given the substantial resources needed for fine-tuning, a reliable DA approach that minimizes hyper-parameter search and ensures favorable outcomes is essential. Additionally, previous complexities attributed to resource constraints, like GPUs and user-friendly frameworks, are no longer valid arguments, enabling the seamless application of intricate processes to crucial tasks such as DA.

We developed a 4-step technique for text classification data augmentation that is time-efficient, stable, intuitive and outperforms existing DA methods. Our experiments with five transformer-based models and four datasets validate our approach, showcasing its superior performance and reliability (Figure 1 and 2).

2 Relevant Studies

In NLP, DA can be challenging due to the contextual nature of the data. Preserving relative word positions is crucial for contextual text embedding, but many existing DA techniques disrupt coherence by introducing random synonyms, punctuations, or altering token order. Regarding ground truth, research falls into two categories: one conserves the original ground truth, while the other generates ground truth based on the augmented sample, with subsequent studies aligning with one of these approaches.

Fadaee et al. (2017) introduced Translation DA for Neural Machine Translation (NMT) by replacing common words with unique words in both source and target sentences. Sennrich et al. (2016) used automatic translation of additional monolingual data for NMT augmentation. Back-translation techniques, as employed by Silfverberg et al. (2017) and Yu et al. (2018), aimed to capture paraphrases for various NLP settings. In addition to EDA (Wei and Zou, 2019), other studies focused on synonym replacements (e.g., Wang and Yang, 2015; Kolomiyets et al., 2011; Zhang et al., 2015). Kobayashi (2018) replaced words with predicted words from BERT, while Andreas (2020) replaced sentence segments with similar contextual counterparts. Sun et al. (2020) used transformers to interpolate input sequences for generating new samples and labels. Additionally, Karimi et al. (2021) compared their work with Xie et al. (2017), viewing it as a data-noising approach to enhance training architectures in NLP.

Many of these approaches, such as AEDA (Karimi et al., 2021) and the work by Xie et al. (2017), often resemble data-noising methods rather than true DA, lacking coherent sentence structures in augmented samples. This falls short of achieving the clarity and human interpretability found in computer vision's approach. To address this, our method extracts coherent, meaningful units from samples, leading to logical samples that surpass existing techniques that disrupt token orders. While most of our experiments focus on text classification due to space constraints, our approach is adaptable to various NLP tasks and holds the potential to become an industry standard.

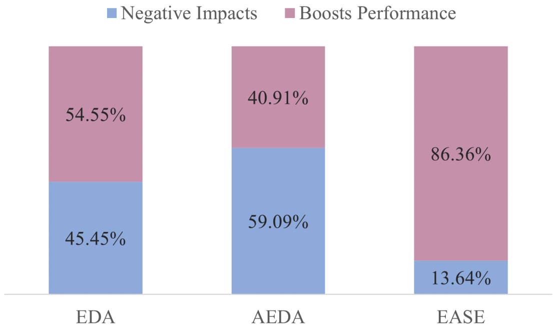


Figure 2: EASE has significantly fewer negative-impacts on performance with different hyper-parameters compared to EDA & AEDA

.

3 EASE

DA with EASE has 4 easy steps that are intuitive and effective.

Extracting Units: In EASE, the most critical step involves extracting meaningful units as augmented samples. The choice of unit depends on the sample structure. For paragraphs, we recommend extracting sentences using the NLTK library (Bird et al., 2009). When dealing with sentences, we suggest extracting "Facts" as introduced by Yuan et al. (2020). These Facts represent coherent sentence units containing logical information. They also preserve token sequences crucial for attention mechanisms in Transformer-based models. For detailed information on extracting facts from sentences, we refer readers to Yuan et al. (2020).

Label Acquisition: In the subsequent EASE step, labels are obtained using pretrained models. The extracted meaningful token sequences make it straightforward for pretrained models to generate high-quality labels without the need for additional training. For our experiments, we employed the

Dataset	Method	Accuracy
Large Movie Review Dataset	Original	86.94%
	EDA	86.72%
	AEDA	86.5%
	EASE	**87.64%**
Sentiment 140	Original	58.67%
	EDA	57.93%
	AEDA	58.23%
	EASE	**60.20%**
Financial Phrase Bank	Original	84.50%
	EDA	**85.20%**
	AEDA	84.72%
	EASE	85.10%
Customer Review	Original	88.55%
	EDA	88.54%
	AEDA	88.32%
	EASE	**89.10%**

Table 1: Comparing EASE, EDA, AEDA for four different datasets in low-resource scenarios by varying the number of augmented samples from small to full size. For Customer Review, the numbers represent average accuracy over three different training subsets with one model, for the other datasets the average is taken over 5 different models and augmentation size variations (Complete detail available in Appendix D). **Bold** suggests the best performance across each column for each dataset.

Dataset	Method	Accuracy
Large Movie Review Dataset	Original	53.37%
	EDA	56.02%
	AEDA	53.08%
	EASE	**67.82%**
Sentiment 140	Original	44.15%
	EDA	44.91%
	AEDA	45.69%
	EASE	**46.00%**
Financial Phrase Bank	Original	55.12%
	EDA	55.35%
	AEDA	55.89%
	EASE	**56.22%**

Table 2: Comparison among EASE, EDA, AEDA for three different datasets in **extremely** low-resource scenarios (only 10 training samples). The performances represent the average over 5 different models (Complete detail available in Appendix D). **Bold** suggests the best performance across each column for each dataset, and parentheses suggest a negative impact on performance)

default pretrained DistilBERT model (fine-tuned on the SST-2 dataset (Socher et al., 2013)) from the HuggingFace library (Wolf et al., 2020) for label generation. In the results section, we present ablation studies to highlight the significance of this step. Nevertheless, it is worth noting that our method can yield promising results even without the label acquisition process.

Sift & Employ : In the "Sift" step, we recommend filtering out smaller-length samples. In our experiments, we retained 10%, 25%, 50%, or 100% of the augmented samples, but it rarely adversely affects performance. This optional step underscores the stability of our method, which is not a random noise injector but a DA technique that complements original training samples. Subsequently, in the "Employ" step, the augmented samples are seamlessly integrated with the original ones completing the final training set.

4 Experimental Setup

We view EDA & AEDA to be the most relevant to our study and showcase performance comparisons for these two methods. Fine-tuning for transformers is usually performed for 5-15 epochs, and from all our experiments, we observe that max validation accuracy is reached before the 30th epoch for these models, but we still performed all the fine-tuning for up to 50 epochs for completeness (More detail on performance saturation in Appendix B). The compared methods differ in augmentation processes: they generate a fixed number of augmented samples per original sample (recommended from 1 to 16), while our approach adapts to sample structure. On average, Fact extraction increases the training dataset by 2.3 times, and sentence extraction by 5.92 times.

4.1 Datasets and Models

For our experiments we used four different sentiment classification datasets. Large Movie Review Dataset (IMDB50K or IMDB) (Maas et al., 2011), Financial Phrasebank (Malo et al., 2014), Customer Review (Hu and Liu, 2004), and Sentiment 140 (Go et al., 2009). We used five different models for our experiments. These are, Bert-base-cased, Bert-base-uncased (Devlin et al., 2019), Distilbert-base-cased, Distilbert-base-uncased (Sanh et al., 2019) and Albert-base-v1 (Lan et al., 2020). We used Huggingface's (Wolf et al., 2020) implementation of these models, a popular Transformer library.

5 Results

5.1 Low-Resource Setting

The original datasets, comprising high number of samples (e.g., 25,000 for IMDB50K, 1.6 Million for Sentiment 140), is adequate for high-performing models like Transformers. To simulate a low-resource scenario, we use only a small subset (e.g., 1000 for Sentiment 140) of the original training sets for data augmentation and generate significantly lower amount of augmented samples compared to EDA & AEDA. (DA) (Details in Appendix D).

In the Sentiment 140 dataset, we have observed that EASE derives benefits from generating augmented samples for the Neutral class, a class that is absent in the original training set but exists in the test set. This stands in contrast to EDA and AEDA. Additionally, EASE demonstrates superior stability and performance.

We observe higher accuracy gain and fewer negative impacts with EASE on average across the board (table 1 & figure 2). Even though, on average, the accuracy gain seems to be higher for EDA, we see the highest accuracy gain of 3.2% in bert-base-cased and fewer negative-impacts with our method for Financial Phrase Bank (Complete table in Appendix D).

Although this study focuses on low-resource scenarios, we still show that our method has promise in high-resource scenarios. Tests on the CR dataset using different portions of the original dataset (500, 2000, and Full) shows that even with the complete dataset, our method outperforms the two other methods, with approximately 10-16x fewer number of augmented samples required (table 1, see Appendix fig. 5 for details).

On an average, we see the best accuracy improvement in 3 out of the 4 datasets with EASE (figure 3). While the other two methods fail to achieve performance boost on an average on 3 out of the 4 datasets, EASE steadily increases performance across all the four different datasets, speaking to the robust nature of our method.

5.2 Extremely Low-Resource Setting

We test the robustness of our method by simulating extremely low-resource scenarios where only 10 training samples are available for fine-tuning and therefore, augmentation. Table 2 demonstrates that even in extremely low-resource setting our method outperforms the other two methods.

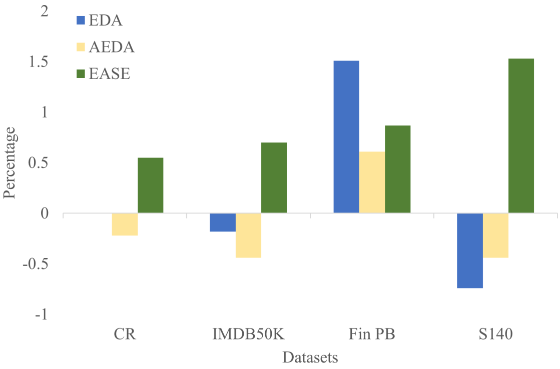


Figure 3: Average accuracy increase over different datasets. EASE showing greater number of and more stable accuracy improvement compared to EDA & AEDA

.

6 Ablation Study

	EASE	EASE-A
Avg. Acc. Gain	1.12%	-0.50%
Neg. Impact	16%	60%
Pos. Impact	84%	40%

Table 3: Average Performance of EASE vs EASE without Acquiring labels on IMDB50K & S140

As an ablation study, we try to measure how important acquiring new labels for the augmented samples is. We use IMDB50K & S140 dataset and test our method by preserving labels. We use the same augmented and original dataset partitions used in typical experiments. The details are summarized in table 3. See Appendix table 8 for details.

7 Conclusion

We introduced an efficient DA technique for TC that improves accuracy without significantly extending training time. Our method outperforms AEDA & EDA in performance, stability, and efficiency. While currently tailored for TC, we envision its adaptation to various NLP tasks with minimal modifications. For instance, equivalent units can be derived from larger samples for Machine Translation using the same technique as EASE to feed the model augmented samples that provide a more nuanced and granular understanding of the training text. Future work will explore additional extraction units and label acquisition methods.

References

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566, Online. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

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

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573, Vancouver, Canada. Association for Computational Linguistics.

William Falcon. 2019. `https://www.pytorchlightning.ai`. Accessed: 2022-08-13.

Alec Go, Richa Bhayani, and Lei Huang. 2009. `http://help.sentiment140.com/home`. Accessed: 2022-08-13.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 168–177, New York, NY, USA. Association for Computing Machinery.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. AEDA: An easier data augmentation technique for text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2748–2754, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. 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 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. 2011. Model-portability experiments for textual temporal analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 271–276, Portland, Oregon, USA. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*. OpenReview.net.

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.

Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4):782–796.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

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.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99, Vancouver. Association for Computational Linguistics.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria. Association for Computational Linguistics.

Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, Philip Yu, and Lifang He. 2020. Mixup-transformer: Dynamic data augmentation for NLP tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3436–3440, Barcelona, Spain (Online). International Committee on 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.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563, Lisbon, Portugal. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

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

Ziang Xie, Sida I. Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y. Ng. 2017. Data noising as smoothing in neural network language models.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension.

Ruifeng Yuan, Zili Wang, and Wenjie Li. 2020. Fact-level extractive summarization with hierarchical graph mask on BERT. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5629–5639, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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.

A Example Augmentations

Table 4 shows two different kinds of augmentated samples with EASE.

Data	Text	Label
	Fact Extraction	
Orig.	The monitor is simply amazing, however, it does not support HDMI input	pos
Aug. 1	The monitor is simply amazing, however,	pos
Aug. 2	it does not support HDMI input	neg
	Sentence Extraction	
Orig.	Actually I'm surprised there were so many comments about this movie. I saw it as part of a Slavic film festival at a major American University. But nobody in USA has heard of it, which is a real shame!	pos
Aug. 1	Actually I'm surprised there were so many comments about this movie.	pos
Aug. 2	I saw it as part of a Slavic film festival at a major American University.	pos
Aug. 3	But nobody in USA has heard of it, which is a real shame!	neg

Table 4: Original sentence and the augmented samples generated and labelled through EASE using Fact or Sentence Extraction.

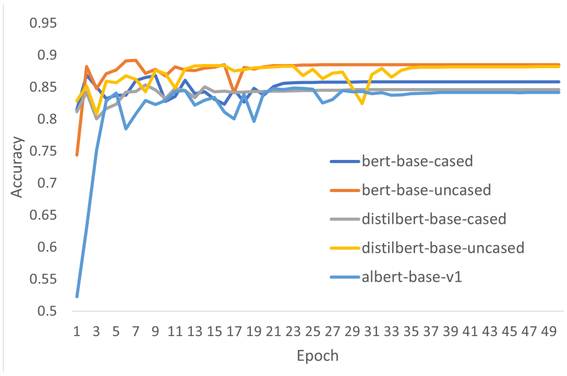


Figure 4: Performance Saturation after 30 epochs for the unaugmented IMDB50K dataset with 500 samples over different models

.

B Performance Saturation

Since the transformer models are already pretrained on unlabelled data, very little amount of fine-tuning is required to gain good task-oriented performance from them. It also must be noted that because of

	Original		EASE			EDA			AEDA		
	Train	Test	Small	Med	Full	Small	Med	Full	Small	Med	Full
CR	500	451	56	282	564	500	2500	4500	500	4000	8000
CR	2000	451	216	1083	2167	2000	10000	18000	2000	8000	32000
CR	4067	451	443	2217	4434	4067	20335	36603	4067	32536	65072
IMDB	500	25000	1000	-	5962	500	-	4500	500	-	4000
FinPB	1000	485	123	-	1235	1000	-	8000	1000	-	9000
S140	1000	497	111	559	1118	1000	5000	9000	1000	4000	8000

Table 5: Number of augmentations used in each experiments for each dataset and each method

the large size of the Transformer based models, even fine-tuning for 50 epochs on multiple GPUs using distributed strategies requires a long time. We discuss more about this in the subsequent section. In all our experiments, we have observed that the validation accuracy in most scenarios saturates after the 30th epoch. In figure 4 we show how fine-tuning for more than 30 epochs is not required. Nevertheless, we still performed all our experiments for 50 epochs for completeness.

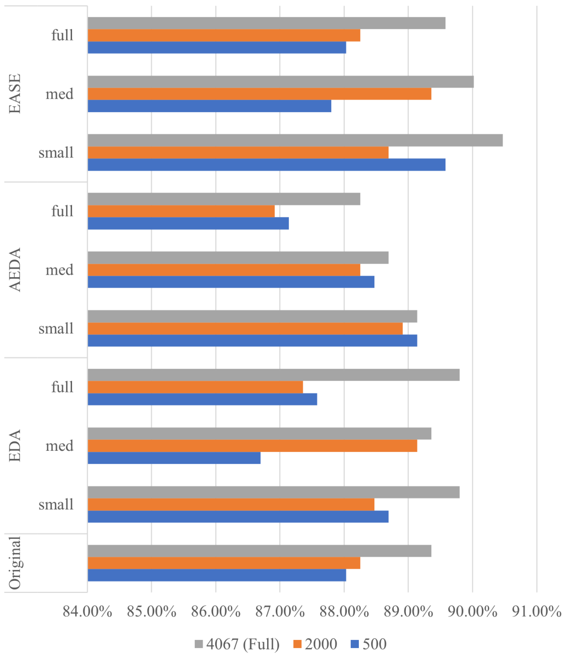


Figure 5: Performance comparison on CR dataset on different training set size using bert-base-cased

.

C Discussion on Training Time

While GPUs are more accessible and distributed training with tools like PyTorch Lightning (Falcon, 2019) has simplified, neural network models are growing larger to balance it out. Transformer models are notorious for taking a massive amount of time for training. To put things into perspective, fine-tuning the Bert-base-cased model for 50 epochs with AEDA-full-augmented IMDB50K dataset (4500 training samples & 25000 testing samples) with 2 Nvidia Tesla P100 GPUs (Each with 16GB Memory) required 12.6 Hours and AEDA-full-augmented Customer Review dataset (65,072 training samples and 451 testing samples) required 26.3 hours. Naturally, searching hyperparameter (number of augmentation) to figure out the optimal augmented dataset that boosts performance is a non-trivial factor to consider while choosing the data augmentation method. For the Customer Review dataset, it took more than 2 days of training to get the results for the different number of augmentation samples, while our method took only 3.4 hours of training. After exploring this vast search space, our method boosted performance 8 out of 9 times, whereas AEDA boosted performance 3 out of 9 times (average performance gain is also in the negative for AEDA). In a low-training-resource scenario, the amount of DA is essential, so a dependable method is required. For these reasons, although our method outperforms EDA & AEDA, we also want to focus on the time-efficient and stable nature of our method.

D Training Set Size and Performance Details

To simulate low-resource settings, small subsets of original training sets were used. Table 5 presents these numbers for each dataset. Model-wise performances are laid out in table 6 for low-resource experiments, in table 7 for extremely-low resource experiments, and in table 8 for the ablation study of label preservation. Customer Review dataset were partitioned into 3 different sets and the accuracy comparisons are showcased in figure 5.

	bert-base-cased	bert-base-uncased	distilbert-base-cased	distilbert-base-uncased	albert-base-v1
Large Movie Review Dataset					
Original	86.92%	89.20%	85.36%	**88.38%**	84.86%
+EDA-small	87.04%	(88.58%)	(85.03%)	(86.84%)	85.17%
+EDA-full	87.26%	(88.22%)	(85.26%)	(87.63%)	86.59%
+AEDA-small	87.31%	(88.98%)	(84.62%)	(87.37%)	85.59%
+AEDA-full	(85.83%)	(88.26%)	(84.98%)	(86.50%)	85.52%
+EASE-small	87.74%	89.40%	85.47%	(87.72%)	86.46%
+EASE-full	**88.01%**	**89.60%**	**86.80%**	(87.72%)	**87.50%**
Sentiment 140					
Original	60.36%	60.56%	59.15%	57.75%	55.53%
+EDA-small	60.56%	61.77%	59.15%	57.95%	56.14%
+EDA-medium	(58.35)%	(58.95%)	(57.34%)	58.15%	(54.12%)
+EDA-full	(57.75)%	(58.55%)	(57.95%)	(57.55%)	(54.73%)
+AEDA-small	(58.95)%	(60.36%)	(58.55%)	58.55%	56.14%
+AEDA-medium	(59.96)%	(60.36%)	(58.35%)	(56.94%)	56.74%
+AEDA-full	(58.15)%	(59.96%)	(56.34%)	57.75%	56.34%
+EASE-small	(59.76)%	**63.18%**	(58.75%)	58.55%	57.75%
+EASE-medium	60.97%	62.17%	**59.96%**	60.36%	**57.95%**
+EASE-full	**62.98%**	62.37%	**59.96%**	**61.97%**	56.34%
Financial Phrase Bank					
Original	84.12%	87.01%	83.71%	84.33%	83.30%
+EDA-small	85.36%	(86.80%)	84.33%	**85.77%**	**84.12%**
+EDA-full	84.95%	87.01%	**85.98%**	85.77%	(81.86%)
+AEDA-small	86.80%	(84.95%)	(83.30%)	84.33%	(83.09%)
+AEDA-full	(84.74%)	**87.84%**	(83.09%)	85.36%	83.71%
+EASE-small	**87.22%**	(84.74%)	83.92%	84.54%	83.92%
+EASE-full	86.19%	87.63%	84.54%	84.95%	83.30%

Table 6: Comparing EASE, EDA, AEDA for the IMDB50K, S140 & FinPB datasets in low-resource scenarios by varying the number of augmented samples from small to full size. **Bold** suggests best performance across each column for each dataset, and parentheses suggest negative-impact on performance

	bert-base-cased	bert-base-uncased	distilbert-base-cased	distilbert-base-uncased	albert-base-v1
Large Movie Review Dataset					
Original	51.85%	54.19%	52.36%	55.97%	52.49%
EDA	54.75%	58.67%	54.41%	61.92%	50.36%
AEDA	(51.52%)	(53.18%)	52.72%	56.11%	(51.91%)
EASE	**64.18%**	**74.85%**	**69.27%**	**72.16%**	**58.66%**
Sentiment 140					
Original	40.24%	60.00%	40.44%	40.04%	40.04%
EDA	(37.22%)	60.00%	45.67%	(37.83%)	43.86%
AEDA	(37.63%)	60.62%	45.88%	(39.64%)	**44.67%**
EASE	**40.44%**	(59.79%)	**46.88%**	**41.05%**	41.85%
Financial Phrase Bank					
Original	59.38%	36.61%	59.38%	60.83%	59.38%
EDA	59.38%	38.83%	59.38%	61.03%	(58.14)%
AEDA	59.38%	**39.64%**	59.38%	61.65%	(59.18)%
EASE	59.38%	38.63%	**61.86%**	**61.86%**	**59.38%**

Table 7: Comparing EASE, EDA, AEDA for the IMDB50K, S140 & FinPB datasets in **extremely** low-resource scenarios (10 Samples) over 5 different models. **Bold** suggests best performance across each column for each dataset, and parentheses suggest negative-impact on performance

	bert-base-cased	bert-base-uncased	distilbert-base-cased	distilbert-base-uncased	albert-base-v1
Large Movie Review Dataset					
Original	86.92%	89.20%	85.36%	**88.38%**	84.86%
+EASE-small	87.74%	89.40%	85.47%	(87.72%)	86.46%
-A	(86.48%)	(88.93%)	(84.30%)	(87.16%)	86.02%
+EASE-full	**88.01%**	**89.60%**	**86.80**%	(87.72%)	**87.50%**
-A	87.13%	89.21%	85.41%	(87.36%)	86.01%
Sentiment 140					
Original	60.36%	60.56%	59.15%	57.75%	55.53%
+EASE-small	(59.76)%	**63.18%**	(58.75%)	58.55%	57.75%
-A	(58.75%)	60.56%	59.36%	59.36%	(54.53%)
+EASE-medium	60.97%	62.17%	**59.96%**	60.36%	**57.95%**
-A	(58.35)%	60.76%	(58.15%)	58.15%	(53.52%)
+EASE-full	**62.98%**	62.37%	**59.96%**	**61.97%**	56.34%
-A	(58.95)%	(59.15%)	(55.73%)	(56.94%)	(54.93%)

Table 8: EASE's performance after preserving labels (EASE vs EASE-A). **Bold** suggests the best performance across each column of each dataset, and parentheses suggest a negative impact on performance.

Enrichment of Arabic WordNet Using Machine Translation and Transformers

Mohamed Dia Eddine Souci [1], Younes Cherifi [1], Lamia Berkani [1], Mohamed Seghir Hadj Ameur [2], Ahmed Guessoum [2]

[1] Faculty of Informatics, USTHB University, Bab Ezzouar, Algiers , Algeria
[2] Higher National School of Artificial Intelligence (ENSIA), Sidi Abdellah, Algiers, Algeria

lberkani@usthb.dz; {mohamed.hadj.ameur; ahmed.guessoum}@ensia.edu.dz

Abstract

This article aims to enhance Arabic WordNet (AWN) by exploiting the current version of the Princeton WordNet (PWN) and Deep Learning (DL) techniques, known for their effectiveness in machine translation. We aim to improve the coverage and quality of AWN by adding new Synsets, integrating definitions and examples, and validating semantic relationships. The contribution can be summarized in three aspects: (1) utilizing multiple translation systems to translate PWN resources and web-extracted data into Arabic; (2) employing Transformers, a highly effective deep learning technique, to refine the outcomes from the first step; and (3) developing a web portal that enables users to visualize proposed updates and facilitates validation by human experts. The results from the evaluation of a random sample of 1,000 Synsets taken from the candidate pool for enrichment are highly promising, with a manual validation accuracy of 75.4%. With such a validation accuracy, the potential would be to get almost 68,000 Synsets correct out of the 90,127 candidates produced by our approach.

1 Introduction

In most of Natural Language Processing applications, the effective handling of semantics relies heavily on the presence of lexical-semantic resources. Among these, WordNets stand out as crucial ones. They can be used in tasks like text classification, information retrieval, and semantic analysis (Morato et al., 2004). WordNets are lexical databases that organize language words based on their meanings. These databases are built on a foundational structure known as the WordNet backbone, which encompasses a hierarchical arrangement of conceptual categories referred to as a taxonomy. Each of these concepts is represented by a set of lemmas (words) that share identical meanings, forming what is known as a Synset. This framework essentially forms a semantic network where words are connected to their corresponding concepts.

The challenge lies in constructing these valuable databases. Although the initial WordNet, known as the Princeton WordNet (Fellbaum, 1998), was meticulously constructed for the English language by linguistics experts and established itself as a standard reference, many other languages lack the resources required to create comparably comprehensive WordNets for various applications. The manual construction and expansion of these WordNets are laborious and resource-intensive tasks. Thus, researchers are striving to devise methods to automate these processes and minimize human involvement.

Our work is primarily based on leveraging PWN, because of its substantial size and extensive coverage, in the enrichment of the AWN. Initially, our approach involves gathering all the grouping lemmas (names) of Synsets along with their existing definitions from PWN. Subsequently, we translated them from English to Arabic using multiple translation engines. To further enrich AWN, recognizing that a majority of PWN Synsets lack examples, we have devised a method to extract a set of relevant examples for each Synset from the web, based on its context. By so doing, we have constructed potential Arabic Synsets, maintaining the same relationships present in PWN. We have obviously taken care of preserving the same hierarchical structure as well as the mapping from PWN. Once these candidate Synsets are generated, a validation process becomes crucial. This step involves manual validation by human experts through a web-based validation portal. Finally, we have extracted the outcomes of the validation step and compiled the newly enriched AWN by incorporating the validated candidate Synsets.

2 Related work

AWN is in a constant state of evolution and expansion, with numerous research efforts aimed at enriching and improving it since its inception in 2006 (Black, et al., 2006).

The initial attempt to enhance AWN was conducted by Alkhalifa and Rodríguez (2018). This work involved utilizing the Wikipedia encyclopedia for the automatic extraction of Arabic named entities that had English equivalents in PWN. Boudabous et al (2013) proposed a linguistic method based on two phases. The first one defines morpho-lexical patterns using a corpus developed from Arabic Wikipedia. The second phase uses patterns to extract new semantic relations from AWN entities. Abouenour et al. (2016) presented an enrichment of AWN targeting three types of content required by Arabic Q-A systems: (1) enrichment of instances or named entities; (2) enrichment of verbs and nouns by extending the list of verb senses and refining the hyponymy relationship between AWN noun Synsets; and (3) enrichment of broken (i.e. irregular) plurals, a class of plural forms that is widely used and precisely defined in Arabic. Hadj Ameur et al. (2017) proposed an automated approach to enrich WordNets sharing the same structural framework as PWN and whose existing Synsets (concepts) are mapped onto the PWN concepts. The authors employed resource-based methods, including dictionaries and ontologies, along with corpus-based methods to extract unambiguous Arabic lemmas and lexical relations. Subsequently, they translated the lemmas into English and paired them with their corresponding PWN lemmas, generating a set of candidate Synsets. Each candidate was assigned a score using a set of features, and these feature parameters were optimized using a suitable metaheuristic. Finally, vocalization and usage examples were provided for each candidate and added to AWN. Batita and Zrigui (2018) focused on enriching antonymy relationships in AWN. Lam et al. (2014) proposed approaches for generating WordNet Synsets for both resource-rich and resource-poor languages, using publicly available WordNets, an automatic translator and/or a single bilingual dictionary. Al Tarouti and Kalita (2016) introduced a novel enrichment approach to enhance the conventional translation approach by utilizing word vectorization (Word Embeddings). This approach involves constructing an initial WordNet in the target language T and enriching it using the approach presented in (Lam et al. 2014). Subsequently, using the Word2vec algorithm, the authors generated word vectors from an existing corpus. These vectors were then used to filter words that belong to each generated Synset, retaining only word pairs with the highest cosine similarity. Utilizing the same similarity measure and existing WordNets such as PWN, a similarity threshold was computed between pairs of synonymous words and semantically related words. This threshold was then used to validate candidate Synsets. Batita and Zrigui (2019) provide a case study on the updates of AWN and the development of its contents, focusing on the relations that have been added to the extended version.

3 Design

In this section, we will explore the intricacies of the approach we have designed to enhance AWN. The process involves several key stages, beginning with the alignment of English and Arabic WordNets and progressing through the extraction of essential elements, including definitions, examples and the translation of words. Ultimately, we will elucidate the final step in this process: the integration of these enriched elements into AWN.

3.1 Alignment with PWN

In order to enrich AWN, we will adopt an alignment-based approach with the Princeton WordNet, which contains the most extensive set of Synsets among all WordNets, with a total of 117,659 Synsets, compared to the 11,269 Synsets of AWN. We establish selection criteria for words extracted from PWN. An English word will be considered a candidate for enrichment if it lacks a corresponding translation in Arabic within AWN, as illustrated in Figure 1. In this way we will have obtained a new, reduced subset of PWN which contains only the Synsets that will be considered in the AWN enrichment operation.

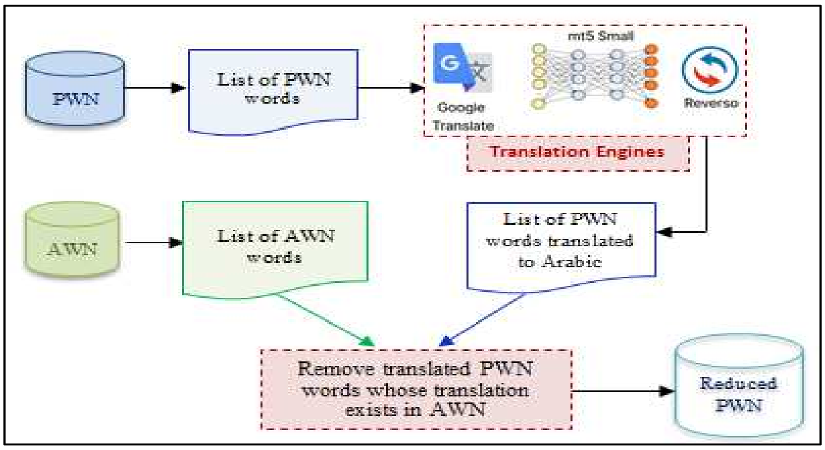


Figure 1: PWN-AWN alignment schema

3.2 Extraction of definitions

Definitions contextualize the word by placing it within its semantic context. We leveraged definitions in our enrichment approach to more accurately determine the meaning of translated words and eliminate any possible ambiguity. This step in our approach involves gathering definitions of words from each Synset that will be integrated into AWN. To accomplish this, we needed a lexical resource that would allow us to collect data automatically. However, we did not find such a resource for the Arabic language. To address this challenge, we decided to use the definitions already present in PWN and directly translate them into Arabic. We followed the steps illustrated in Figure 2 and described below it.

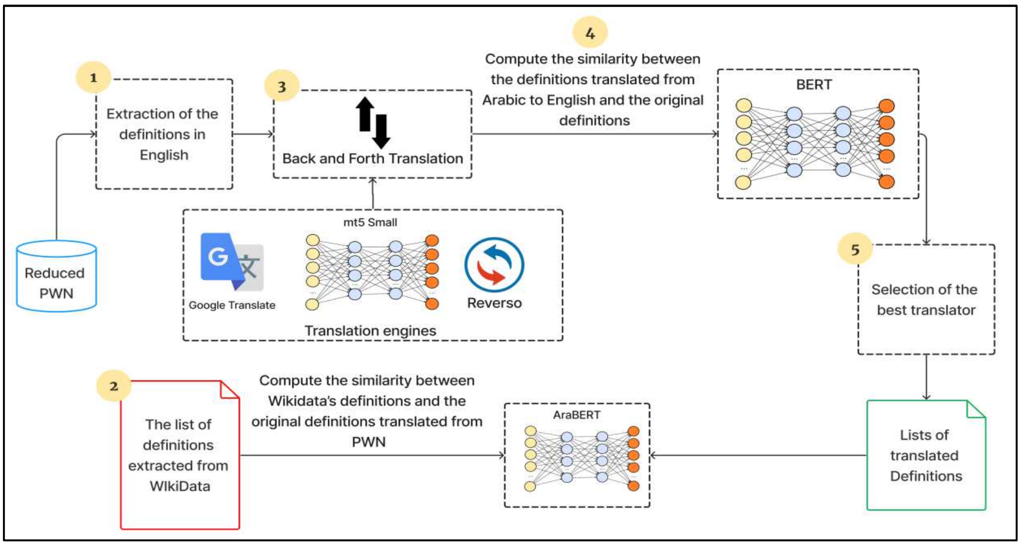

1
Extraction of the definitions in English
3
Back and Forth Translation
4
Compute the similarity between the definitions translated from Arabic to English and the original definitions
BERT
Reduced PWN
mt5 Small
Google Translate
Reverso
Translation engines
5
Selection of the best translator
2
The list of definitions extracted from WIkiData
Compute the similarity between Wikidata's definitions and the original definitions translated from PWN
AraBERT
Lists of translated Definitions

Figure 2: Extraction of definitions

1) We proceeded by retrieving, for each Synset in PWN, its English definition. Then we translated this into Arabic using various translation systems, including Google Translator, Reverso, and our custom translation model developed using the Transformer MT5.

2) Additionally, we extracted other definitions from Wikidata. However, after manually analyzing some of these definitions, we observed that the majority of them lacked precision in terms of the original context of the Synset. Nevertheless, we opted to retain them as candidate definitions, with the intention of getting them through automated evaluation at a later stage.

3) Following the compilation of lists of candidate definitions, we employed the Back-And-Forth Translation logic to compare the translated definitions with the original definitions and evaluate their similarities to determine the

most effective translation system among the aforementioned three.

4) To assess the similarity while considering the context of the definitions, we utilized the BERT language model to extract Word Embeddings and calculate cosine similarity between the translations of the definitions and their originals.

5) By comparing the original definitions with the translated candidate definitions, we selected the translation that maximized the similarity. This way we made sure that the translated definitions are as close as possible to the original definitions, while preserving context and meaning.

6) Finally, we generated a list of candidate definitions that provides a range of translated definitions tailored to context, accurately representing the meanings of the Synsets.

3.3 Word Translation

To generate the list of Arabic words translated from the names of English Synsets, we followed the same steps as those applied for the extraction of definitions (see Figure 3):

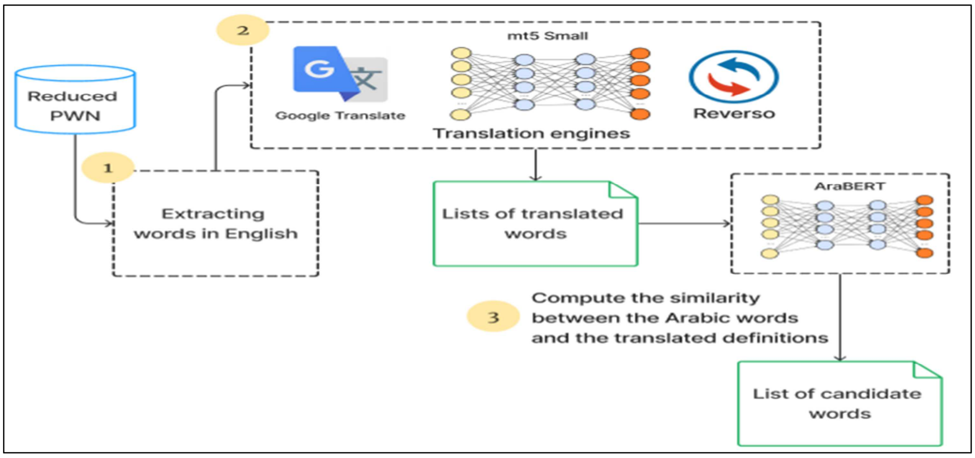

Reduced PWN
1
Extracting words in English
2
mt5 Small
Google Translate
Reverso
Translation engines
Lists of translated words
AraBERT
3
Compute the similarity between the Arabic words and the translated definitions
List of candidate words

Figure 3: Schema representing the process followed to translate and select words

The steps followed for the translation of words are described as follows:

1) First, we extracted words from the names of the Synsets available in PWN.

2) Similar to the approach used for definitions, we found it preferable to utilize a variety of translation tools such as Google Translate and Reverso Translate to obtain multiple candidate translations for each Synset. This was necessary because these translators do not consider the context of the word during translation.

3) After obtaining the translations of each word from the PWN Synsets, we evaluated their similarities with the translated Arabic definitions using the AraBERT model and the cosine similarity measure. This step enabled us to retain only the candidate word translations that exhibited maximum contextual similarity.

4) Finally, we generated a list of translated words by selecting the best candidates that correspond to the different meanings and contexts of the Synsets.

3.4 Extraction of Examples

Examples are highly valuable in WordNets as they provide concrete illustrations of word usages, associated senses, and the context in which a word would appear. To maximize the number of examples for each Synset, we followed the steps outlined below, as illustrated in Figure 5.

We utilized Web Scraping techniques to extract relevant data from the Reverso website, which provides examples covering various contexts and uses for most words in the Arabic language. The extracted data were saved in a text file to be used in the following steps. However, we found that not all the extracted examples were correct, i.e. some of them contained Latin characters or were simply in English. Therefore, a preprocessing was performed to eliminate these incorrect examples using a model we trained by fine-tuning the AraBERT model.

After generating the final list of examples for each Synset, we evaluated the similarity of each example with the previously chosen Arabic definition in the first step using the AraBERT model. Hence, by selecting examples that exhibited maximum similarity with the Arabic definition, we ensured that we chose an example closely aligned with the specific meaning and context of the Synset.

3.5 Validation Module

After generating a final list of candidates for each Synset (its translation, definition, and a list of contextual examples), this list was submitted to a web validation platform. This platform enables human experts to manually review each Synset. Experts could confirm the accuracy of the models used, identify potential errors or inconsistencies in the proposed Synsets, and make corrections.

3.6 Insertion into AWN

The final phase of our approach will focus in the future on generating the XML file containing the newly validated Synsets for integration into the current version of AWN, which is available on the GlobalWordNet portal.

We are optimistic about the acceptance of the vast majority of the generated Synsets since the validation platform will allow human experts to correct any potential errors that may be detected in terms of translation or inappropriate examples or definitions for a given Synset word. We expect that the number of retained words after filtering may be reduced in cases where the translation of the word and/or definition is rejected. If the semantic relations of a translated Synset are validated, we will include them in the XML file while maintaining the same hierarchical structure as that of PWN. However, if any semantic relation is not valid, we will add it as an independent Synset, meaning that it will not be linked to other Synsets in the hierarchy of the newly enriched AWN.

4 Experiment

This section covers the technical aspects related to the development of our approach, including the tools used. It also provides illustrations of each step of our approach. Finally, it will present a summary of the results obtained, along with an analysis and discussion.

4.1 Results of word extraction

To accomplish this task, we have made use of the WordNet library from the nltk.corpus package. We employed the wordnet.all_synsets () function, which returns a list containing all 117,659 Synsets from PWN 3.0. Subsequently, for each Synset, we retrieved its identifier using the synset.name () function. Each identifier has the format illustrated in Figure 4:

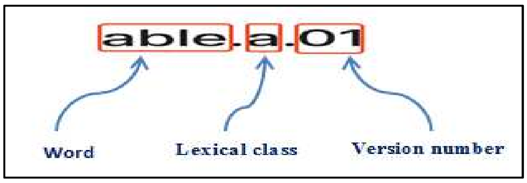
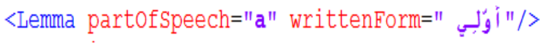


Figure 4: Synset ID Format

Finally, using Python programming we retrieve the word as the first part of the Synset ID.

We followed the same procedure for AWN, which contains 60,157 Synsets in its second version, "arb2-lmf," available for download from GlobalWordNet [1] website. This version of AWN is provided in XML format. We extracted words from the "writtenForm" attribute of the <Lemma> tag, as follows:



```
<Lemma partOfSpeech="a" writtenForm=" أوّلي "/>
```

Therefore, after retrieving the two lists of words, we retained only the words from PWN that do not have translations in AWN. Table 1 represents the number of extracted words:

WordNet	Number of extracted words
PWN	117,659
AWN	19,978
Reduced PWN	90,127

Table 1: Number of extracted words.

4.2 Choosing the best translation system

As previously mentioned, we used the Back-And-Forth Translation logic to select the translation system that provides the best Arabic translations. To achieve this, we opted for employing the original definitions from PWN as a reference for this task. Initially, we translated each definition from English to Arabic using three translation systems: Google Translate, Reverso, and our MT5 model.

Next, we back-translated the produced (Arabic) results into the English language. Subsequently, we employed cosine similarity to compare these back-translated definitions with the original English definitions. The translation

[1] http://globalwordnet.org/

system that yielded translations most closely aligned with the meaning and context of the original English definitions was chosen as the optimal system for translation. This approach ensures that the selected translation system produces as accurate and contextually relevant translations as possible for the enrichment of AWN.

In the following, we present two graphs that highlight the similarities between different translated definitions and the original definitions. The first graph in Figure 5 illustrates the similarities of a sample of 100 definitions translated with Google Translate and the MT5 model. The second graph in Figure 6 shows the similarities of the same sample of definitions translated with Google Translate and Reverso.

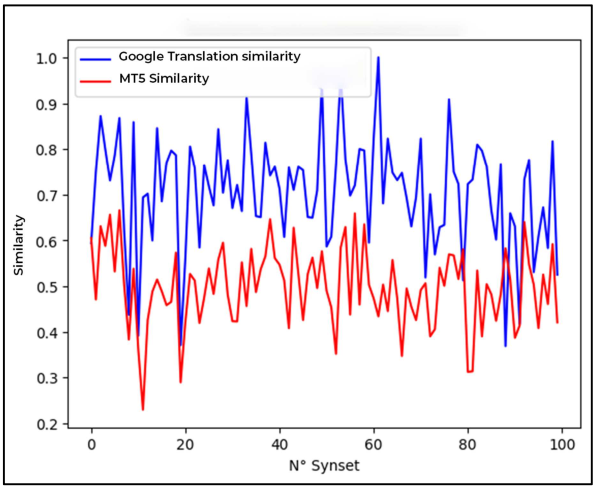


Figure 5: Comparison of translation similarities between Google Translate and MT5

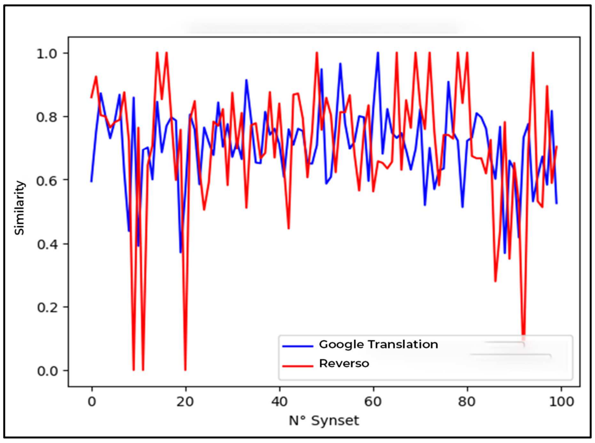


Figure 6: Comparison of translation similarities between Google Translate and Reverso

Analyzing the first graph, we concluded that Google Translate produces a much better translation quality compared to the MT5 model. Similarities are generally higher with Google Translate. On the other hand, the second graph indicates that Google Translate and Reverso provide translation quality that is almost similar, with only a few exceptions. This is why we have chosen to use Google Translate as the primary translator for the step of translating PWN definitions.

4.3 Definition filtering

The purpose of this step is to choose the best source of definitions between Wikidata and PWN. We used Google Translate to translate the definitions extracted from Wikidata into English. Then, we used BERT to calculate their similarities with the original definitions from PWN. The following graph highlights the similarities between the definitions from Wikidata, the translated definitions from PWN, and the original definitions from PWN.

The results show that the definitions extracted from Wikidata are not precise enough to capture the original context of the Synset. In fact, the similarities of the Wikidata definitions were at most 65%, whereas the translated definitions had similarities ranging from 65% to 100%, for the vast majority of these definitions. We thus chose to use the translated definitions for integration into the enriched AWN.

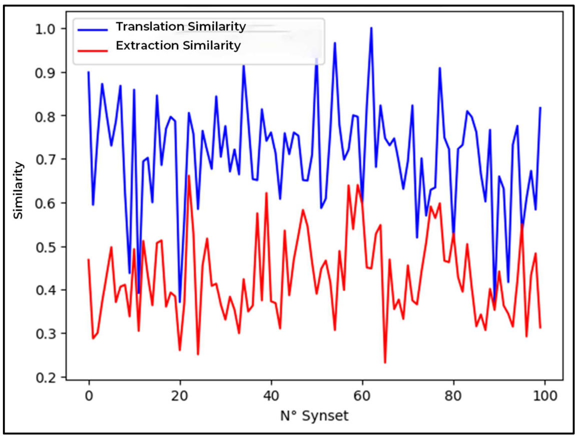


Figure 7: Similarity comparison between PWN and Wikidata

4.4 Example filtering

This step comprises two essential parts: (1) preprocessing the extracted data, and (2) selecting the examples to integrate into AWN.

Preprocessing data: As mentioned in the previous chapter, we used Reverso to extract the list of examples in Arabic (around 20 examples for each Synset translated from PWN). However, after manually analyzing some instances, we noticed the presence of several poorly formed and incorrect examples. To address this issue, we utilized the fine-tuned AraBERT model that we trained to classify sentences as either correct or incorrect. The results obtained were extremely satisfactory, demonstrating an outstanding performance of the classification model. However, despite achieving a high accuracy of 98% during model training, we observed that the lack of data led to a few misclassifications. Nevertheless, these errors did not pose a major obstacle to the next step. With the extraction of multiple examples for each translated word, we managed to compensate for any potential confusion. This approach allowed us to improve the quality of the examples and significantly reduce the size of the selected candidate examples, thus paving the way for more accurate and reliable results.

Selecting examples: After performing an initial filtering of the examples, we proceeded to the second step, which involves selecting the best examples, those that maximize contextual similarity with the senses of the Synsets translated into Arabic. For this purpose, we leveraged the AraBert model, which enabled us to calculate the similarity between the filtered examples and the definitions from PWN translated into Arabic using Google Translate. Once this step was completed, we selected, for each Synset, the most relevant example among all candidates. Table 2 represents some examples and their similarities with the definitions:

Synset ID	Definition in Arabic	Examples in Arabic	Similarity
able.a.01	يتبعها عادةً ("إلى") امتلاك الوسائل أو المهارة أو الدراية أو السلطة اللازمة للقيام بشيء ما	يعتقد البعض أن البنوك المركزية هي وحدها القادرة على ذلك	0.79
		من الناحية المثالية، سيكون لديك شريك قادر للمساعدة في التخطيط والتنفيذ	0.82
unable.a.01	يتبعها عادةً ("إلى") لا تمتلك الوسائل أو المهارة أو المعرفة اللازمة	وقال إن حكومة إسلام أباد تبدو وكأنها عاجزة أو رافضة للسيطرة على أراضيها	0.75
		وعندما تركت الكرسي المتحرك لاحظت أنها لا تستطيع استخدام الجانب الأيسر من جسدها	0.76

Table 2: Examples and their similarities with the definition

4.5 Word filtering

Similar to the selection of examples, we followed the same steps to choose the most appropriate translations for the sense of each Synset. Table 3 represents some words and their similarities with the definitions.

Synset ID	Definition in Arabic	Words in Arabic	Similarity
able.a.01	يتبعها عادةً ("إلى") امتلاك الوسائل أو المهارة أو الدراية أو السلطة اللازمة للقيام بشيء ما	إمكانية	0.47
		قادر	0.16
		جدير	0.15
unable.a.01	يتبعها عادةً ("إلى") لا تمتلك الوسائل أو المهارة أو المعرفة اللازمة	غير قادر	0.56
		لا تستطيع	0.52
abaxial.a.01	تواجه بعيدًا عن محور العضو أو الكائن الحي	محور	0.22

Table 3: Some words and their similarities with the definition

4.6 Construction of a Candidate Synset

Once we generated the lists containing the candidate translations (words, examples, definitions), we proceeded to create the XML file that would group the 90,127 candidate Synsets ready to be integrated into our validation platform database. Table 4 illustrates the obtained results:

Number of candidate Synsets	90,127
Number of candidate words added	231,165
Number of candidate examples added.	297,190
Number of candidate definitions added.	90,127
Number of Synsets with examples.	78,498
Number of words considered on average for each Synset	3
Average number of examples considered for each Synset.	3
Number of definitions considered for each Synset.	1

Table 4: Statistics of candidate Synsets.

4.7 Evaluation of the Results

In this section, we will delve into the results of the validation process conducted on a subset of 1,000 Synsets. Obviously, the validation is intended to be carried out by linguistics experts (and it will be done in the near future). However, in order to have a first assessment of the potential quality of our approach, we took on the responsibility of the manual evaluation of the sample. We consulted both the Collins English dictionary and the Elmaany Arabic dictionary to cross-reference the original English Synsets with their translated counterparts. Our validation process was facilitated through a dedicated platform. For word validation, we meticulously checked the English words, their contexts, and compared them with the translated Arabic versions. When it came to definitions, the majority were straightforward, allowing for a direct comparison between the English and Arabic definitions. In the case of examples, we meticulously selected the Arabic examples corresponding to the context of each word and proceeded with their validation.

Out of the randomly selected sample of 1000 Synsets, we obtained the following results:

	Validated number	**Rejected number**	**Precision**
Words	789	211	**78.9 %**
Definitions	952	48	**95.2 %**
Examples	813	187	**81.3 %**

Table 5: Precision of validation

5 Conclusion

In this work, we have presented a substantial enrichment of AWN along with a validation mechanism. Our contribution involved extending the currently available version of AWN on Global WordNet with automatically constructed Synsets based on PWN. We retrieved PWN Synsets, filtered out existing AWN equivalents, and translated them into Arabic, creating a list of candidate Synsets. Additionally, we extracted context-based examples for each Synset to enrich AWN. Deep learning techniques, particularly Transformers, were employed to evaluate and filter the Arabic Synset candidates. The results from the evaluation of a random sample of 1,000 Synsets taken from the candidate pool for enrichment are highly promising, with a manual validation accuracy of 75.4%. This work can further be enriched using other sources than just PWN, like Arabic text corpora. One could also consider using WordNets from other languages and look for ways of improving the quality of the translation.

References

Jorge Morato, Miguel Ángel Marzal, Juan Lloréns, and José Moreiro. 2004. In Proceedings of the Global WordNet Conf., GWC 2004, Petr Sojka, Karel Pala, Pavel Smrž, Christiane Fellbaum, Piek Vossen (Eds.), pages 270–278.

Christiane Fellbaum. 1998. WordNet: An Electronic Lexical Database, MIT Press. ed.

Horacio Rodríguez, David Farwell, Javi Farreres, Ma nuel Bertran, Musa Alkhalifa, Antonia Marti, William Black, Sabri Elkateb, James Kirk, Adam Pease, Piek Vossen, and Christiane Fellbaum. 2008. Arabic WordNet: Current State and Future Extensions. In Proceedings of the 4h Global WordNet Conf., Szeged, Hungary, pages 387–405.

William Black, Sabri Elkateb, Horacio Rodríguez, Musa Alkhalifa, Piek Vossen, Adam Pease, Manuel Bertran, and Christiane Fellbaum. 2006. The Arabic WordNet Project. In Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06).

Musa Alkhalifa, and Horacio Rodríguez. 2018. Automatically Extending Named Entities coverage of Arabic WordNet using Wikipedia. International Journal on Information and Communication Technologies, 3(3): 20-36.

Mohamed Mahdi Boudabous, Nouha Chaâben Kammoun, Nacef Khedher, Lamia Hadrich Belguith, and Fatiha Sadat. 2013. Arabic wordnet semantic relations enrichment through morpho-lexical patterns. In Proceedings of the first International Conf. on Communications, Signal Processing, and their Applications, pages 1–6.

Mohamed Seghir Hadj Ameur, Ahlem Chérifa Khadir, and Ahmed Guessoum. 2017. An Automatic Approach for WordNet Enrichment Applied to Arabic WordNet. In International Conf. on Arabic Language Processing, pages 3–18.

Mohamed Ali Batita, and Mounir Zrigui. 2018. The Enrichment of Arabic WordNet Antonym Relations. In: Gelbukh, A. (eds) Computational Linguistics and Intelligent Text Processing. CICLing 2017. LNCS, 10761, Springer, Cham.

Khang Nhut Lam, Feras Al Tarouti and Jugal Kalita. 2014. Automatically Constructing WordNet Synsets. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 106–111, Baltimore, Maryland, USA, June 23-25 2014.

Feras Al Tarouti and Jugal Kalita. 2016. Enhancing Automatic Wordnet Construction Using Word Embeddings. In Proceedings of the Workshop on Multilingual and Cross-lingual Methods in NLP, pages 30–34, San Diego, California. Association for Computational Linguistics.

Mohamed Ali Batita, and Mounir Zrigui. 2019. The Extended Arabic WordNet: a Case Study and an Evaluation Using a Word Sense Disambiguation System. In Proceedings of the 9th Global WordNet Conference.

Compiling a Corpus of Technical Documents for Dialogue System Development in the Industrial Sector

Laura García-Sardiña,[1], Eneko Ruiz,[2] Cristina Aceta,[3]
Izaskun Fernández,[3] María Inés Torres,[2] Arantza del Pozo,[1]
[1]Vicomtech Foundation, Basque Research and Technology Alliance (BRTA)
[2]University of the Basque Country UPV/EHU, Speech Interactive Research Group
[3]Tekniker Foundation, Basque Research and Technology Alliance (BRTA)

Abstract

This work deals with the compilation of a machine-readable corpus of technical documents ready to be processed for the development of data-driven dialogue system applications in the industrial sector. To this end, we propose a pipeline to convert technical *PDF* documents into a set of *JSON* files that allow machine processing of their information. This procedure is able to extract and organise a variety of content types such as text, images, and tables in order to obtain a corpus of structured information, which additionally allows easy conversion into other formats for further visualization or processing purposes. A qualitative analysis of the proposed procedure by expert technical operators has resulted in a positive validation of the proposal. The compiled sample includes Question Answering annotations and instances of the dialogue ontology related to industrial procedures that shall allow the development of voice user interfaces to assist technical operators dealing with industrial tasks.

1 Introduction

The emergence of virtual assistants, mainly in the leisure and domestic scopes, proves that interacting through natural language with different devices and applications is a reality that keeps improving. In this context, the demand for such capacities in industry is increasing, since natural communication with industrial systems leads to productivity and security improvements which reduce operation time and costs (González-Docasal et al., 2021), providing operators 4.0 with powerful and intuitive interaction mechanisms to perform their tasks successfully (Romero et al., 2020).

In fact, the concept of human-centric manufacturing where a collaborative intelligence assists operators in their needs (Lu et al., 2022) has recently been introduced, replacing more traditional system-centric factories. In order to reduce the cost of developing such systems, machine learning based methods are increasingly being considered (Torres, 2013; Zamora et al., 2017), as they are easier to develop and maintain. However, these systems require considerable amounts of data for development (Wang et al., 2018), and one of the main caveats in industrial settings is the lack of corpora for model training (Serras et al., 2020; Vázquez et al., 2023; Justo et al., 2010). A common practice to obtain data for under-resourced scenarios is to exploit available documentary records from the domain at hand (Tiedemann, 2014). In industrial scenarios, common relevant sources are technical documents such as user manuals and manufacturing and assembly dossiers, which are usually available in PDF format.

Automatic information extraction and structuring from PDF documents is a difficult unsolved task (Bast and Korzen, 2017), with scarce previous work, especially in the industrial domain. Documented approaches (Dong et al., 2021) exploit PDF technical reports with relatively simple structures and only consider text extraction, leaving aside the preservation of images and tables, which are especially frequent and relevant in industrial technical documents.

This work proposes a pipeline to convert industrial technical documents in PDF format into a machine-readable JSON structure, that can be used to develop data-driven dialogue system applications. In particular, annotations for Question Answering (QA) and procedure-related instances for the development of ontology-based Dialogue Systems (DS) have been compiled. QA systems allow obtaining information from a collection of unstructured documents (Wang et al., 2021; Xingguang et al., 2022), and are evolving towards conversational interfaces (Reddy et al., 2019), whereas ontology-based DS allow to define domains in detail and reduce ambiguity between agents (Antonelli and Bruno, 2017), leading to structured

Manufacturer	Machines	Tasks
ABB	IRB120	Maintenance
Stäubli	Robotic arm	Maintenance
Fanuc	F11 and F12	CNC
Fagor	8055, 8060	CNC
Fagor	8065, 8070	CNC
Siemens	PG 1106	CNC
Heidenhain	TNC 426	CNC
Heidenhain	TNC 430	CNC
Ikor	RACK 81.51	Assembly

Table 1: Source technical manuals.

knowledge representations required in procedural assistance applications. The combination of these technologies shall enable the development of voice-based assistants aimed at guiding operators throughout maintenance tasks, providing answers to technical questions and/or assisting operators in procedures extracted from industrial technical documents.

The rest of the paper is structured as follows. Section 2 introduces the main characteristics of industrial technical documents and a machine-readable corpus structure for them. Section 3 describes the proposed pipeline to convert technical documents into the that format, while the annotation procedure for relevant question-answer pairs is presented in Section 4. Next, Section 5 provides details regarding the corpus compiled using the proposed pipeline, as well as some evaluation metrics. Finally, conclusions are drawn in Section 6.

2 Industrial Technical Documents

Industrial documents of technical nature cover a wide range of topics. In this work, we have selected a set of 9 documents, focusing on three relevant industrial tasks in which operators used to frequently consult technical documents when interacting with industrial production machines and processes, and where the development of QA systems as well as procedural assistants, along with spoken interfaces, can have considerable productivity impact: (a) maintenance, (b) machine programming, and (c) manufacturing and assembly tasks.

Table 1 details the selected documents from different manufacturers, spanning from robot maintenance manuals, or computer numerical control (CNC) programming manuals to manufacturing and assembly dossiers, all in PDF format.

Despite the structures of the manuals fluctuate across manufacturers, all documents include headers and footers, chapter and sub-chapter based levels, as well as a considerable amount of tables and images embedded in the text. In addition, tables and images usually include key information that cannot be overlooked for QA and DS applications for industrial procedure assistance, e.g. they often describe processes or show parts of the machine where actions are required. For instance, Figure 1 shows a PDF fragment including an image indicating the concrete positioning of a particular element, to be considered during mechanical failure reparations or maintenance activities.

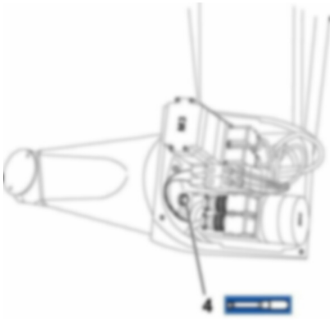


Figure 1: Fragment of a manual showing an image with a pointer to a relevant position for a maintenance task. The image has been intentionally blurred to comply with copyright restrictions.

So, relevant information for QA and dialogue for assistance can correspond to components of various sizes from an entire section or subsection, to a short span of text, or even be a table or a part of it, or an image. Thus, it is of key importance that converted documents are structured in a way that preserves all this information in a machine-readable format.

In order to enable dialogue applications to exploit the structural information of the source industrial technical documents in PDF format, the following machine-readable corpus structure has been defined:

- **Manual**. Each manual gives rise to a dialogue use case, so there is a folder for each manual.
- **Chapters**. The contents of the manuals are divided into chapters, so there is a folder for each chapter.
- **Sections**. Chapters are often divided into sections, so a JSON file is created for each section, comprising its contents in a structured way. If the chapter has no sections, a single file is created for the chapter's contents. Each

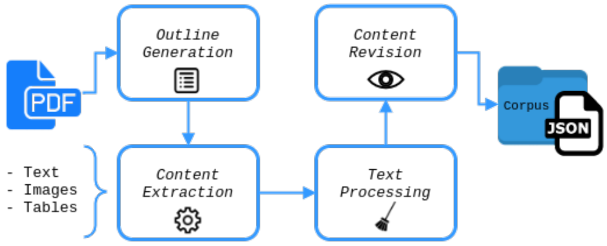


Figure 2: Main steps of the corpus compilation pipeline.

chapter folder also contains an *images* folder where the images extracted are stored.

- **Subsections**. Sections can be further divided into subsections, when they appear as part of the numbered structure (e.g., 4.1.1 is a subsection of section 4.1 in chapter 4), or what we called ***titles***, which are distinguished parts of the contents with their own name but which do not follow the numbered schema. We also consider as *title* the level below *subsection* when it exists, numbered or not.

- **Contents** in the deepest level. When the deepest level of a manual's structure is reached for each element of the nested listings (e.g., a subsection with no title elements as children), its contents are incorporated as a dictionary structure including: the raw text; the processed (clean) text; the images found for the excerpt, as a list of the names with which they have been stored; the list of tables that appear in the excerpt; and the final text after manual revision and corrections.

3 PDF Document Conversion Pipeline

The pipeline proposed to automatically transform the source industrial technical PDF documents into the machine-readable corpus structure introduced in Section 2 and depicted in Figure 2, consists of four main steps:

1. **Outline generation.** This outline is a structured representation in JSON format of the different levels of contents conforming the document.

2. **Content extraction.** This step consists of extracting the contents in the original manuals, using the outlines produced in step one to correctly locate fragments, and saving them in a structured form. This step includes the extraction of three types of contents of interest: text, images, and tables.

3. **Text processing.** The raw text extracted from the manuals is processed to produce a clean version containing only the text in the main body, free of undesired segments such as page headers and footers, text fragments that belong inside tables, or contents belonging to other fragments that have not been correctly delimited in the extraction step.

4. **Content Revision.** An optional revision can be carried out manually to ensure that the final contents of the documents are correct.

Detailed information about the tools that were tested and those that were finally integrated into the pipeline as well as on the implementation of the different steps of the pipeline are presented in the next subsections.

3.1 Outline Generation

PDF documents with a valid navigation-enabling outline, preserve information about their internal structure (i.e., chapters, sections, subsections, etc. and their starting pages) in their metadata. Exploiting this information enables to generate the base JSON structure of the outline in an automatic way.

The `pdftohtml` (Glyph & Cog, 2021b) PDF conversion utility of the `xpdf` (Glyph & Cog, 2021d) toolkit was used to extract the PDF's metadata relative to the contents outline in a structured HTML format that preserves the elements' hierarchy, titles and start pages. The result can be parsed to extract the relevant information and format it into the targeted JSON structure. The library used to parse the HTML information was `AdvancedHTMLParser` (Savannah, 2021). The fragments' end page numbers, which are not present in the PDF files outlines, are established by some heuristics and assumptions.

3.2 Content Extraction

There exists a variety of tools aimed at automatically extracting content from PDF files. However, after trying several, none of them seemed to be optimal for extracting all three targeted types of contents, text, images, and tables. For this reason, different methods were tested and selected for the extraction of each content type.

The methodology used for testing and selecting the optimal tools for content extraction involved the following steps: 1) Applying each tool on a selected manual; 2) Analysing the produced result; 3) If the result was unsatisfactory the tool would be discarded; 4) The tools that performed best would be applied to other manuals to check whether the results were consistent on other documents from different manufacturers, and so deem them as acceptable.

3.2.1 Text Extraction

The following tools available which are oriented to the extraction of text from PDF documents were tested for our pipeline:

- `pdfplumber` (Singer-Vine, 2021). Although this library is more oriented to the extraction of tables from PDF files, its text extraction functionality was also tested. Results were bad, given that text spacing was not interpreted correctly by the extractor and words appeared joined together.

- `tika-python` (Mattmann, 2021) is a library that makes use of the Apache Tika toolkit[1] for extracting text from PDF. Results of applying this library across manuals were generally good, although there were some cases where page numbers appeared joined together with the main text, without spacing.

- `pdftotext` (Glyph & Cog, 2021c) is `xpdf`'s command line utility to extract text from PDF files. As no errors were detected when applying it across the manuals, this tool was selected and incorporated in the pipeline for text extraction.

3.2.2 Images Extraction

We found only a few tools capable of extracting images from PDF files. Some included little to no documentation on how to use this functionality. For example, `pdfplumber` extracts some image representation objects from the PDF's metadata, but it does not provide information on how to obtain the original images from those representations. The image extractors that produced valid results when applied to our manuals are:

- `pdfimages` (Glyph & Cog, 2021a) is a `xpdf`'s command line utility to extract images from PDF files. Images across manuals were correctly extracted and saved into a target folder. Errors were found with some images' formats.

- `PyMuPDF` (Jorj X. McKie and Liu, 2021) along with `Pillow` (Clark, 2021). The combination of both Python libraries allows extracting and saving images from PDF files. Results were comparable to the previous tool, some errors being still encountered with certain images. This method was selected and incorporated to the pipeline for image extraction.

3.2.3 Tables Extraction

As for table extraction, we only found a couple of tools aimed at extracting tables from PDF files. General content extractors such as the already-mentioned `pdftohtml` would not recognise tables. The tools specialised in table extraction that we tested are:

- `pdfplumber` (Singer-Vine, 2021): Despite this library is specifically oriented to extract tables from PDF files, the results of its application were bad. For some tables, only the header row was recognised as a table, and their text contents included spacing errors. Many other tables were simply not recognised as such.

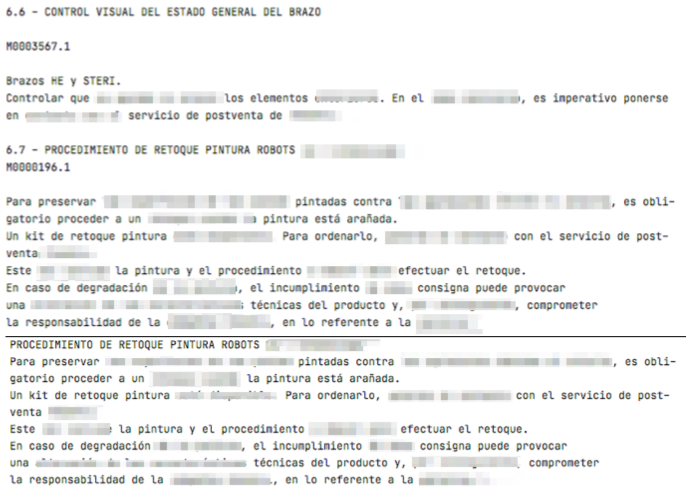
- `camelot` (Mehta, 2021) is also a Python library specifically aimed at extracting tables from PDF documents, and it allows saving the extracted tables as JSON, among other formats. Results of applying this tool across manuals were mostly good, with a few cases of tables not automatically recognised as such. This was the tool selected and incorporated into the pipeline for table extraction.

Given that table extractors work at page level, a common consequence is that tables that span over several pages are always considered as individual, separate tables. Since in the manuals it is usual for tables to repeat the header row when continuing in a new page, a processing function was created to automatically join split tables when necessary.

3.3 Text Processing

This step of the pipeline aims to delete undesired information. Within text fragments this mainly includes headers and footers, page numbers, text fragments that belong inside tables, and content that has not been correctly delimited and does not belong to the current fragment. In some cases, it also includes expressions that do not provide meaningful information and hinder readability of the

[1] http://tika.apache.org/



```
6.6 - CONTROL VISUAL DEL ESTADO GENERAL DEL BRAZO

M0003567.1

Brazos HE y STERI.
Controlar que [redacted] los elementos [redacted]. En el [redacted], es imperativo ponerse
en [redacted] servicio de postventa de [redacted]

6.7 - PROCEDIMIENTO DE RETOQUE PINTURA ROBOTS [redacted]
M0000196.1

Para preservar [redacted] pintadas contra [redacted], es obli-
gatorio proceder a un [redacted] pintura está arañada.
Un kit de retoque pintura [redacted] Para ordenarlo, [redacted] con el servicio de post-
venta [redacted]
Este [redacted] la pintura y el procedimiento [redacted] efectuar el retoque.
En caso de degradación [redacted], el incumplimiento [redacted] consigna puede provocar
una [redacted] técnicas del producto y, [redacted], comprometer
la responsabilidad de la [redacted], en lo referente a la [redacted]
```

```
PROCEDIMIENTO DE RETOQUE PINTURA ROBOTS [redacted]
Para preservar [redacted] pintadas contra [redacted], es obli-
gatorio proceder a un [redacted] la pintura está arañada.
Un kit de retoque pintura [redacted] Para ordenarlo, [redacted] con el servicio de post-
venta [redacted]
Este [redacted] la pintura y el procedimiento [redacted] efectuar el retoque.
En caso de degradación [redacted], el incumplimiento [redacted] consigna puede provocar
una [redacted] técnicas del producto y, [redacted] comprometer
la responsabilidad de la [redacted], en lo referente a la [redacted]
```

Figure 3: Above, text as extracted in step 2. Below, same text after applying the text processing step. Contents have been intentionally blurred to comply with copyright restrictions.

corresponding fragment (e.g. “Continued on next page" or alphanumeric codes used by the manufacturer, such as “M000..." in Figure 3).

Regular expressions are used to delete headers, footers, page numbers, and undesired information in general. They are fed from a configuration file adapted to each manual format, as it changes across industrial documents. Although some of the employed regular expressions are simple and straightforward, others get more complex to avoid deleting meaningful information inside the text. Figure 3 shows an example of a text fragment in which the footer, page number, and an alphanumeric code that deteriorates readability have been erased.

The process used to automatically delete content not belonging to a particular text fragment consists of the following steps: 1) The level of the current element is identified (section 6.7 in Figure 3); 2) Any text appearing before the title of the current element is deleted (in the example, text belonging to section 6.6 is deleted); 3) The title of the next element is identified and all text from that point on, if included, is erased from the current fragment.

3.4 Quality Revision

The final step of the pipeline involves an optional manual revision process in order to ensure the quality of the final corpus.

In order to make this process easier for human reviewers, each fragment to be reviewed is dumped into a text file following this format: (i) A header including the fragment’s starting and ending page in the PDF, plus the fragment’s title and a separator; (ii) The processed text as obtained from the previous step of the pipeline, and a separator; and (iii) A preview of the tables found for the current fragment, formatted as text using the `tabulate` (Astanin, 2021) library on the JSON table data. The first and third parts are provided as helping references, while the part appearing between separators is the one to be reviewed and possibly modified. This part is then loaded and included in the final JSON after revision.

3.5 Pipeline in use

The presented PDF document conversion pipeline has been applied and qualitatively validated extracting the information contained in the technical manuals listed in Table 1. Generally speaking, structuring and automatic content extraction have been quite satisfactory. This impression was confirmed at the content revision step, where three technical reviewers agreed that the information extracted was correct to a great extent. In addition, four expert technical operators have also provided a very positive opinion of the pipeline output, validating it for annotation and information presentation purposes within the project.

The outline generation module was capable of extracting an accurate content structure from the PDF files’ metadata. The selected content extraction tools were capable of extracting raw text, images, and tables without major problems across the varied set of analysed industrial technical documents. And the implemented text processing module allowed to adequately filter out undesired (e.g., headers and footers), irrelevant (e.g. “Continued on next page"), and duplicated information auto-

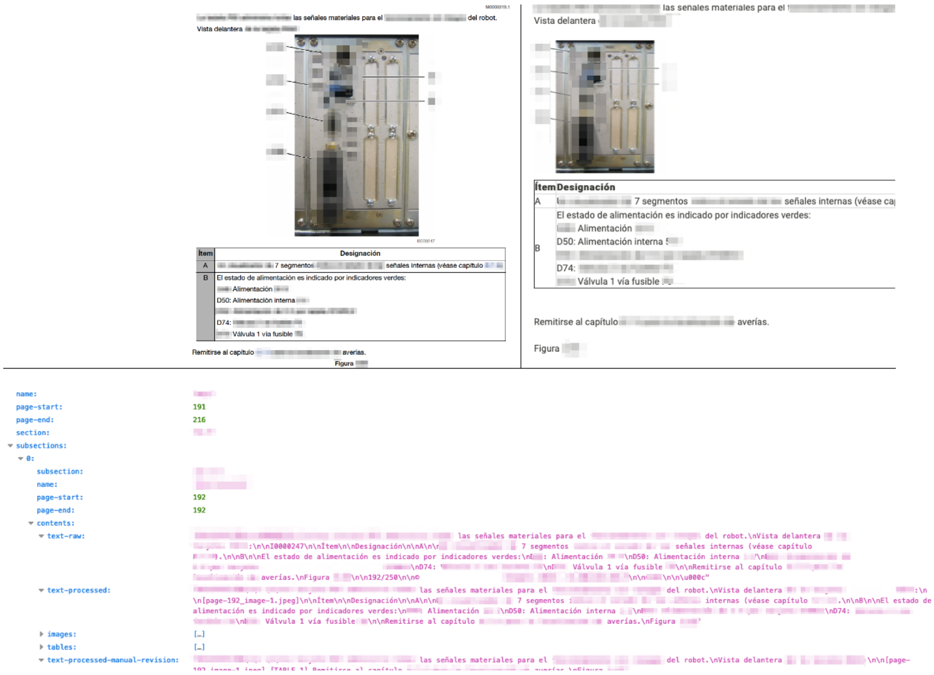


Figure 4: A sample fragment of a manual in its original PDF format (top left), the structured JSON produced by our pipeline (bottom), and recomposed for visualisation on an annotation tool (top right). Contents have been intentionally blurred to comply with copyright restrictions.

matically.

In a few occasions, some content extraction anomalies were detected, such as *missing text content*, which was mostly caused by the missing text being wrongly considered part of a table, and *additional text content*, the origin of which was mostly related with some subsections and their parent sections' names being equal, causing content delimiters to fail. Furthermore, additional content was also generated when tables were incorrectly extracted as text or text from images was considered as part of the main body of text.

Tables, although generally well-extracted, presented more problems than text content. The most common issues were related to *content* (empty, missing, or incomplete tables and multiple tables that were incorrectly joined) and *format* (misplaced cell content or moved content when cells had itemised or numbered lists).

All in all, the conversion pipeline is able to produce a comprehensive version of the original contents in PDF to a machine-exploitable structured JSON format.

4 Annotation Procedure

Once converted, the industrial technical documents were annotated by four skilled operators, in order to identify relevant question-answer pairs and procedures to develop both question-answering and dialogue systems for procedure assistance, respectively. The employed annotation tool (Justo et al., 2016) requires the input to be formatted as Markdown text, which allows inline HTML code. Knowing this, processed text was easily converted from the original JSON files and displayed by the tool with minimal changes (e.g. adapting line breaks). Nevertheless, references to images and tables still had to be converted from JSON to their HTML equivalents to be visualized.

Figure 4 depicts a fragment from a PDF manual, including text, an image, and a table, at three different stages: On the top left, the fragment is shown in its original PDF format; The bottom image illustrates the results of applying the PDF conversion pipeline, where contents are presented as a structured JSON; Finally, the image on the top right shows how the same contents are displayed by the annotation tool.

Two types of question-answer pairs were identified by annotators: Generic and Specific. Generic questions usually describe a procedure (e.g., disassembling part of a robot) and corresponded to whole subsections or, depending on the manual, sub-subsections. On the other hand, answers to specific questions were extracted from either subsections or sub-subsections and generally referred

Manual	# Chapters	# Running Words	Vocabulary size	Norm. Lev. distance
IRB120	3	3167	807	0.66
Controller	2	8368	1883	0.78
Robotic Arm	2	2329	780	0.93
F11	8	23302	3650	0.93
F12	5	37974	4862	0.70
CNC 8060/8065	15	61730	4975	0.91
PG 1106	11	46430	6794	0.81
TNC 426/TNC 430	12	65410	6270	0.86
RACK AA.81.51.2001	9	5019	1525	0.91

Table 2: Basic information about the number of chapters and words per manual as well as the normalized Levenshtein between the manually corrected text and the automatically corrected text.

to particular queries (e.g., what tools should I use to disassemble the part). In total and during six months, the four operators annotated 1416 generic and 751 specific question-answer pairs.

5 Corpus Description and Evaluation

Table 2 provides a general description of the obtained corpus in terms of number of chapters, number of running words and vocabulary size. In order to gain a better insight into the performance of the conversion pipeline we compare the manually corrected text with the text extracted by the system. To this end we considered the Levenshtein distance, which calculates the minimum number of changes, i.e. adding, deleting, or replacing a single character, that are needed to transform one string into the other.

In this way, we can evaluate how well the automatic text extractor and text processor based on regular expressions work. However, it heavily depends on the length of both strings to be compared and does not provide any meaningful information in our case, where the length varies significantly from one manual to the other. To solve this problem, we have adapted it to the normalized Levenshtein distance, defined in Equation 1

$$Lev_{norm} = 1 - \frac{Lev(s1, s2)}{max(len_{s1}, len_{s2})} \qquad (1)$$

where $Lev(s1, s2)$ is the Levenshtein distance between the strings $s1$ and $s2$ and len_{s1} and len_{s1} are the length of the strings. Thus, the distance does not longer depends on the length of both strings. The value of the normalized Levenshtein varies between 1.0, where both strings are equal, and 0.0, where they are completely dissimilar. Table 2 incldues the normalised Levenshtein distance between the correct text with the one extracted through the proposed PDF conversion pipeline. This table shows a very good performance of the extraction and processing tool for most of the manuals, being higher than 0.78 for almost all of them. However, the improper operation of the extraction and processing tools lead to worse performance for both the IRB120 and F12 manuals. These manuals have numerous images, then the extraction tool retrieved information embedded in the images.

On the other hand, Table 3 shows the main characteristics of the QA corpus obtained after the annotation procedure described in Section 4. This table provides the number of generic (GQ) and specific (SQ) questions along with the average lengths of each manual, in terms of the number of words. This table shows a high variability in the length of the answers within the same manual, and from one manual to another. This fact, along with that some answers include images, shows the complexity of developing question-answering systems for the industrial environment. This complexity, for example, does not exist in other more popular extractive question-answer datasets such as SQuAD v1.1 (Rajpurkar et al., 2016) and v2.0 (Rajpurkar et al., 2018), where images are not included and the length of the answers is less variable. However, this dataset has been successfully used for a QA system, which has been tested in the framework of the same research project (Ruiz et al., 2023).

Finally, Table 4 provides information about the corpus for ontology-based DS, detailing instances derived from the 6 selected procedures in the corpus, represented according to the ontology described in (Aceta et al., 2022). From the total of 268 instances, 6 correspond to procedures, 8 to methods, 20 to tasks and 69 to steps. In a nut-

Manual	# Generic questions (GQ)	Avg. # words in GQ	% answers with images	# Specific question (SQ)	Avg. # words in SQ
IRB120	419	1342.2	0	159	101.1
Controller	133	2528.3	0	53	112.3
Robotic Arm	55	1285.7	0	36	88.1
F11	192	2452.9	26	62	120.7
F12	186	2030.9	35	144	182.5
CNC 8060/8065	124	2342.8	21	60	209.7
PG 1106	91	5667.9	27	47	194.1
TNC 426/430	164	2071.4	43	63	210.2
RACK AA.81.51.2001	52	1075.8	48	127	94.7

Table 3: Annotated QA dataset, which includes informations of General (GQ) and Specific Questions (SQ)

Class	Instances	Avg. instances per procedure	Avg. instances per upper-class	Additional info		Descriptions	
				Avg.	SD	Avg.	SD
Procedure	6	-	-	3.66	2.71	1	0
Method	8	1.33	1.33 (procedure)	0.75	1.39	1.63	4.6
Task	20	3.33	2.5 (method)	0.5	1.15	0.35	0.49
Step	69	11.5	3.45 (task)	0.52	0.61	0.49	0.68

Table 4: Ontology instantiation overview

shell, each procedure has an average of 1 method, each method an average of 3 tasks and each task an average of 4 steps.

The rest of the instances in the dialogue system ontology instantiation include other details such as the necessary materials to perform the procedure given a specific method or the descriptions and additional information for each procedure division class. As for the latter, Table 4 also shows the average numbers of instances covering each type of information per each class and the Standard Deviation (SD) to provide more objective insights on this data. As it can be seen in the table, the number of instances may differ significantly depending on the procedure or method.

6 Conclusions and Future Work

This work has been developed in the framework of a research project whose objective is to facilitate industrial maintenance, programming, manufacturing, and assembly tasks through the use of voice-based interfaces. Unlike classic dialogue systems in which the information to be supplied to the user is naturally structured in a database (e.g., restaurants, types of food, addresses, etc.), human-machine interaction assisting industrial operators' tasks needs to handle information that is usually found in PDF documents.

In this scenario, we have proposed a pipeline aimed at extracting content from technical PDF documents and converting them into a machine-readable format required for the automatic processing of their information. This procedure is able to extract and organise a variety of content types such as text, images, and tables in order to get a corpus of structured information organised in JSON files.

A qualitative analysis of the results of the proposed procedure has been carried out by expert technical operators, who have provided a very positive opinion validating the proposal. Moreover, normalised Levenshtein distance between the manually corrected text and the text generated by the pipeline is quite high showing a very good performance of the pipeline. This way, we are contributing to the scarce literature addressing multimodal content extraction from documents in PDF format, which is still mainly limited to text extraction.

In addition, one of the advantages of the proposed procedure to structure contents into JSON files is that it can be easily converted into other formats, such as HTML, for further visualisation or processing purposes. We have taken advantage of this feature to facilitate an annotation process carried out by expert technical operators. These annotations as well as the derived ontology instances have allowed the compilation of a useful corpus for the development of question answering and assistance-oriented dialogue system applications in the industrial sector.

Acknowledgements

This work was supported by the Basque Government's Elkartek research and innovation program, through the EKIN KK-2020/00055) and BERREKIN (KK-2022/00102) projects

References

Cristina Aceta, Patricia Casla, Izaskun Fernandez, and Aitor Soroa. 2022. Kide4assistant: an ontology-driven dialogue system adaptation for assistance in maintenance procedures. In Kyritsis D. Sarkar A. Sanfilippo E.M., Karray M., editor, *Proceedings of the 12th International Workshop on Formal Ontologies Meet Industry (FOMI 2022) Co-Located with Workshops About the Industrial Ontology Foundry (IOF) and the European Project OntoCommons (EU H2020 Project). Tarbes, France, September 12-15*, volume 3240. CEUR Workshop Proceedings.

Dario Antonelli and Giulia Bruno. 2017. Human-Robot Collaboration Using Industrial Robots. In *2017 2nd International Conference on Electrical, Automation and Mechanical Engineering (EAME 2017)*, pages 99–102. Atlantis Press.

Sergey Astanin. 2021. python-tabulate. https://pypi.org/project/tabulate/. Accessed: August, 2023.

Hannah Bast and Claudius Korzen. 2017. A benchmark and evaluation for text extraction from pdf. In *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 1–10.

Alex Clark. 2021. Pillow. https://pypi.org/project/Pillow/. Accessed: August, 2023.

Zihui Dong, Shiladitya Paul, Karl Tassenberg, Geoff Melton, and Hongbiao Dong. 2021. Transformation from human-readable documents and archives in arc welding domain to machine-interpretable data. *Computers in Industry*, 128:103439.

Glyph & Cog. 2021a. pdfimages – Portable Document Format (PDF) image extractor. https://www.xpdfreader.com/pdfimages-man.html. Accessed: August, 2023.

Glyph & Cog. 2021b. pdftohtml – Portable Document Format (PDF) to HTML converter. https://www.xpdfreader.com/pdftohtml-man.html. Accessed: August, 2023.

Glyph & Cog. 2021c. pdftotext – Portable Document Format (PDF) to text converter. https://www.xpdfreader.com/pdftotext-man.html. Accessed: August, 2023.

Glyph & Cog. 2021d. Xpdf – PDF viewer and toolkit. https://www.xpdfreader.com/. Accessed: August, 2023.

Ander González-Docasal, Cristina Aceta, Haritz Arzelus, Aitor Álvarez, Izaskun Fernández, and Johan Kildal. 2021. Towards a natural human-robot interaction in an industrial environment. In Luis Fernando D'Haro, Zoraida Callejas, and Satoshi Nakamura, editors, *Conversational Dialogue Systems for the Next Decade*, pages 243–255. Springer Singapore, Singapore.

Jorj X. Jorj X. McKie and Ruikai Liu. 2021. PyMuPDF. https://pypi.org/project/PyMuPDF/. Accessed: August, 2023.

Raquel Justo, J Alcaide, and M Torres. 2016. Crowdzientzia: Crowdsourcing for research and development. *Proceedings of IberSpeech*, pages 403–410.

Raquel Justo, Oscar Saz, Vîctor Guijarrubia, Antonio Miguel, M. Inês Torres, and Eduardo Lleida. 2010. Improving dialogue systems in a home automation environment. ICST.

Yuqian Lu, Hao Zheng, Saahil Chand, Wanqing Xia, Zengkun Liu, Xun Xu, Lihui Wang, Zhaojun Qin, and Jinsong Bao. 2022. Outlook on human-centric manufacturing towards industry 5.0. *Journal of Manufacturing Systems*, 62:612–627.

Chris A. Mattmann. 2021. tika-python. https://github.com/chrismattmann/tika-python. Accessed: August, 2023.

Vinayak Mehta. 2021. Camelot: PDF Table Extraction for Humans. https://camelot-py.readthedocs.io/. Accessed: August, 2023.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

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.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

D. Romero, J. Stahre, and M. Taisch. 2020. The Operator 4.0: Towards Socially Sustainable Factories of the Future. *Computers & Industrial Engineering*, 139.

Eneko Ruiz, María Inés Torres, and Arantza del Pozo. 2023. Question answering models for human–machine interaction in the manufacturing industry. *Computers in Industry*, 151:103988.

Tim Savannah. 2021. AdvancedHTMLParser. `https://pypi.org/project/AdvancedHTMLParser/`. Accessed: August, 2023.

Manex Serras, María Inés Torres, and Arantza Del Pozo. 2020. Improving Dialogue Smoothing with A-priori State Pruning. In *ICPRAM*, pages 607–614.

Jeremy Singer-Vine. 2021. pdfplumber. `https://github.com/jsvine/pdfplumber`. Accessed: August, 2023.

Jörg Tiedemann. 2014. Improved Text Extraction from PDF Documents for Large-Scale Natural Language Processing. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 102–112. Springer.

M. Inés Torres. 2013. Stochastic bi-languages to model dialogs. In *Proceedings of the 11th International Conference on Finite State Methods and Natural Language Processing*, pages 9–17, St Andrews, Scotland. Association for Computational Linguistics.

Alain Vázquez, Asier López Zorrilla, Javier Mikel Olaso, and María Inés Torres. 2023. Dialogue management and language generation for a robust conversational virtual coach: Validation and user study. *Sensors*, 23(3).

Jinjiang Wang, Yulin Ma, Laibin Zhang, Robert X. Gao, and Dazhong Wu. 2018. Deep learning for smart manufacturing: Methods and applications. *Journal of Manufacturing Systems*, 48:144–156. Special Issue on Smart Manufacturing.

Tian Wang, Jiakun Li, Zhaoning Kong, Xin Liu, Hichem Snoussi, and Hongqiang Lv. 2021. Digital twin improved via visual question answering for vision-language interactive mode in human–machine collaboration. *Journal of Manufacturing Systems*, 58:261–269. Digital Twin towards Smart Manufacturing and Industry 4.0.

Liu Xingguang, Cheng Zhenbo, Shen Zhengyuan, Zhang Haoxin, Meng Hangcheng, Xu Xuesong, and Xiao Gang. 2022. Building a question answering system for the manufacturing domain. *IEEE Access*, 10:75816–75824.

Mauricio Zamora, Eldon Caldwell, Jose Garcia-Rodriguez, Jorge Azorin-Lopez, and Miguel Cazorla. 2017. Machine Learning Improves Human-Robot Interaction in Productive Environments: A Review. In *IWANN2017: Advances in Computational Intelligence*, pages 283–293, Cham. Springer International Publishing.

The 6th International Conference on Natural Language and Speech Processing (ICNLSP 2023)

December 16-17, 2023

All times are according to (GMT)

	Saturday, Dec. 16, 2023 08:00 – 18:10 (GMT)
08:00-08:30	*Opening session* **Dr. Mourad Abbas**
08:30 – 09:10	*Keynote 1*: **Transcending Communication Barriers: From Machine Translation to Language Transparence** **Prof. Alex Waibel**, CMU, USA
09:30 – 11:50	*Oral Session 1: Classification and clustering* **Chair**: Dr. Abed Alhakim Freihat, *University of Trento*, Italy
09:30 – 09:45	**Classification of Human- and AI-Generated Texts for English, French, German, and Spanish** Kristina Schaaff; Tim Schlippe; Lorenz Mindner (IU International University of Applied Sciences)
09:50 – 10:05	**Handling Realistic Label Noise in BERT Text Classification** Maha Agro, Hanan Aldarmaki (Mohamed Bin Zayed University of Artificial Intelligence)
10:10 – 10:25	**Discourse Relations Classification and Cross-Framework Discourse Relation Classification Through the Lens of Cognitive Dimensions: An Empirical Investigation** Yingxue Fu (University of St Andrews)
10:30 – 10:45	**Representation Learning for Hierarchical Classification of Entity Titles** Elena Chistova (FRC CSC RAS)
10:50 – 11:05	**DAP-LeR-DAug: Techniques for enhanced Online Sexism Detection** Jayant Panwar; Radhika Mamidi (IIIT Hyderabad)
11:10 – 11:25	**CommunityFish: A Poisson-based Document Scaling With Hierarchical Clustering** Sami Diaf (Universität Hamburg)
11:30 – 11:45	**ADCluster: Adaptive Deep Clustering for Unsupervised Learning from Unlabeled Documents** Arezoo Hatefi; Xuan-Son Vu; Monowar Bhuyan; Frank Drewes (Umeå University)
11:50 - 13:00	**Break**

13:00 – 16:00	*Oral Session 2: Deep learning and transformers* **Chair**: Dr. Mohammed Mediani, *United Arab Emirates University*, UAE
13:00 – 13:15	**Efficient Black-Box Adversarial Attacks on Neural Text Detectors** Vitalii Fishchuk (University of Twente); Daniel Braun (University of Twente)
13:20 – 13:35	**Transformer-Based Analysis of Sentiment Towards German Political Parties on Twitter During the 2021 Election Year** Nils Constantin Hellwig (Media Informatics Group, University of Regensburg); Markus Bink (Media Informatics Group, University of Regensburg); Thomas Schmidt (Media Informatics Group, University of Regensburg); Jakob Fehle (Media Informatics Group, University of Regensburg); Christian Wolff (Regensburg University)
13:40 – 13:55	**"Japan's Answer to Mozart": Automatic Detection of Generalized Patterns of Vossian Antonomasia** Michel Schwab (Humboldt-Universität zu Berlin); Robert Jäschke (Humboldt-Universität zu Berlin); Frank Fischer (Freie Universität Berlin)
14:00 – 14:15	**GAVI: A Category-Aware Generative Approach for Brand Value Identification** kassem sabeh (Free University of Bozen Bolzano); Mouna Kacimi (Wonder Technology Srl); Johann Gamper (Free University of Bozen-Bolzano)
14:20 – 14:35	**Simple, Simpler and Beyond: A Fine-Tuning BERT-Based Approach to Enhance Sentence Complexity Assessment for Text Simplification** Lucía Ormaechea (University of Geneva); Nikos Tsourakis (University of Geneva); Didier Schwab (University of Grenoble-Alpes); Pierrette Bouillon (University of Geneva); Benjamin Lecouteux (University Grenoble Alpes (UGA))
14:40 – 14:55	**Deep Learning-Based Claim Matching with Multiple Negatives Training** Anna Neumann (Ruhr-University Bochum); Dorothea Kolossa (Ruhr-Universität Bochum); Robert M Nickel (Bucknell University)
15:00– 15:15	**Exploring BERT Models for Part-of-Speech Tagging in the Algerian Dialect: A Comprehensive Study** Mohamed Amine CHERAGUI (Ahmed Draia University); Abdelhalim Hafedh DAHOU (GESIS - Leibniz Institute for the Social Sciences); Amin ABDEDAIEM (Ahmed Draia University)
15:20 – 15:35	**A Neural Network Approach to Ellipsis Detection in Ancient Greek** Giuseppe G. A. Celano (Leipzig University)
15:40 – 15:55	**AraBERT and mBert: Insights from Psycholinguistic Diagnostics** BASMA SAYAH (Amar Telidji); Attia Nehar (Ziane Achour University); Hadda Cherroun (Université Amar Telidji); Slimane Bellaouar (University of Ghardaia)
16:00 - 16:10	**Break**
16:10 – 18:10	*Oral Session 3: Analysis, summarization, and numerical representation* **Chair**: Prof. Maria Inés Torres, *University of the Basque Country*, Spain
16:10 – 16:25	**An NLP Analysis of ChatGPT's Personality Simulation Capabilities and Implications for Human-centric Explainable AI Interfaces** Thorsten Zylowski (University of Hohenheim); Matthias Wölfel (Karlsruhe University of Applied Sciences)
16:30 – 16:45	**Topically diversified summarization of customer reviews** Florian Carichon; Gilles Caporossi (HEC Montreal)
16:50 – 17:05	**Extracting higher-order logic formulas from English sentences** Alexandre Rademaker; Guilherme Lima; Renato Cerqueira (IBM)

17:10 – 17:25	**A Quantitative Approach to Understand Self-Supervised Models as Cross-lingual Feature Extracters** Shuyue Stella Li (Johns Hopkins University); Beining Xu (Beihang University); Xiangyu Zhang (University of New South Wales); Hexin Liu (Nanyang Technological University); Wenhan Chao (Beihang University); Paola Garcia (Johns Hopkins University)
17:30 – 17:45	**Def2Vec: Extensible Word Embeddings from Dictionary Definitions** Irene Morazzoni (DEIB, Politecnico di Milano); Vincenzo Scotti (DEIB, Politecnico di Milano); Roberto Tedesco (DEIB, Politecnico di Milano)
17:50 – 18:05	**Exploring Hybrid Linguistic Features for Turkish Text Readability** Ahmet Yavuz Uluslu (University of Zurich)
	Sunday, Dec. 17, 2023 08:30 – 16:00 (GMT)
08:30 – 09:10	*Keynote 2:* **Biosignal-based Digital Biomarkers for Aging** **Prof. Najim Dehak,** Johns Hopkins University, USA
09:30– 13:40	*Oral Session 4: Speech and phonetics* **Chair**: Prof. Tim Schlippe, *IU International University of Applied Sciences*, Germany
09:30 – 09:45	**Comparison of Wav2vec 2.0 Transformer Models for Speaker Change Detection** Zbyněk Zajíc (University of West Bohemia); Marie Kunešová (University of West Bohemia)
09:50 – 10:05	**Typological classification of European Portuguese fricatives: a cross-language forced alignment and pronunciation variants study** Anisia Popescu (Université Paris Saclay - LISN); Lori Lamel (CNRS LISN); Ioana Vasilescu (LIMSI)
10:10 – 10:25	**Methods for Phonetic Scraping of Youtube Videos** Adrien Meli (Université Paris Cité); Steven Coats (University of Oulu); Nicolas Ballier (Université Paris Cité)
10:30 – 10:45	**Direct Speech to Text Translation: Bridging the Modality Gap Using SimSiam** Balaram Sarkar (Indian Institute of Technology Indore); Chandresh K Maurya (IBM Research); Anshuman Agrahri (IIT, Indore)
10:50 – 11:05	**Improving Dhivehi Automatic Speech Recognition (ASR) with Sub-word Modelling, Language Model Decoding and Automatic Spelling Correction** Arushad Ahmed (University of St. Andrews)
11:10 – 11:25	**Comparing Modular and End-To-End Approaches in ASR for Well-Resourced and Low-Resourced Languages** Aditya Parikh (Radboud University); Louis ten Bosch (radboud unversity); Henk van den Heuvel (Radboud University); Cristian Tejedor-Garcia (Radboud University Nijmegen)
11:30 – 11:45	**Towards Joint Modeling of Dialogue Response and Speech Synthesis based on Large Language Model** Xinyu Zhou (Communication University of China); Delong Chen (HKUST); Yudong Chen (Communication University of China)
11:45 - 13:00	**Break**
13:00 – 13:15	**Using Whisper LLM for Automatic Phonetic Diagnosis of L2 Speech, a Case Study with French Learners of English** Nicolas Ballier (Université Paris Cité); Adrien Meli (Université Paris Cité); Maelle Amand (Université de Limoges); Jean-Baptiste Yunès (Université Paris Cité)
13:20 – 13:35	**Enhancing Word Discrimination and Matching in Query-by-Example Spoken term detection with Acoustic Word Embeddings**

	Pantid Chantangphol (Kasikorn Business Technology Group (KBTG); Theerat Sakdejayont (Kasikorn Business Technology Group (KBTG)); Tawunrat Chalothorn (Kasikorn Business Technology Group (KBTG))
13:40 – 15:40	*Oral Session 5: Dataset* **Chair:** Dr. Daniel Braun, *University of Twente*, Netherlands
13:40 – 13:55	**Turkish Native Language Identification** Ahmet Yavuz Uluslu (University of Zurich)
14:00 – 14:15	**KMD: A New Kurdish Multilabel Emotional Dataset For the Kurdish Sorani Dialect** Soran SM Badawi (Language Center Charmo University)
14:20 – 14:35	**iTANONG-DS : A Collection of Benchmark Datasets for Downstream Natural Language Processing Tasks on Select Philippine Languages** Moses L. Visperas; Christalline Joie Borjal; Aunhel John M Adoptante (DOST-ASTI); Danielle Shine R. Abacial (Mindanao State University - Iligan Institute of Technology); Ma. Miciella Decano (Far Eastern University); Elmer C Peramo (DOST-Advanced Science and Technology Institute)
14:40– 14:55	**Data Augmentation for Text Classification with EASE** A M Muntasir Rahman (New Jersey Institute of Technology); Wenpeng Yin (Penn State University); Guiling Wang (New Jersey Institute of Technology)
15:00 – 15:15	**Enrichment of Arabic WordNet Using Machine Translation and Transformers** Mohamed Dia Eddine Souci, Younes Cherifi, Lamia Berkani (University of Science and Technology Houari Boumediene, Algeria); Mohamed Seghir Hadj Ameur, Ahmed Guessoum (The National Higher School of Artificial Intelligence, Algeria)
15:20– 15:35	**Compiling a Corpus of Technical Documents for Dialogue System Development in the Industrial Sector** Laura García-Sardiña (Vicomtech); Eneko Ruiz (Universidad del País VAsco UPV/EHU); Cristina Aceta (Tekniker); Izaskun Fernández (Tekniker); Maria Inés Torres (Universidad del País Vasco UPV/EHU); Arantza del Pozo (Vicomtech)
15:40	*Closing session*

N.B: TIME IN GMT

Keynote speakers

Prof. Dr. Alexander Waibel
Carnegie Mellon University, USA
Karlsruhe Institute of Technology, Germany
Zoom Research Fellow

Alex Waibel

Alexander Waibel is Professor of Computer Science at Carnegie Mellon University (USA) and at Karlsruhe Institute of Technology (Germany). He is director of the International Center for Advanced Communication Technologies. Waibel is known for work in AI, Machine Learning, Multimodal Interfaces and Speech Translation Systems. He introduced consecutive and simultaneous speech translation in 1991 and 2005. Waibel proposed early Neural Network learning methods, including the TDNN, the first shift-invariant ("convolutional") Neural Net (1987) and many multimodal interaction systems. Waibel founded/co-founded more than 10 startups, including Jibbigo, first speech translator on a phone (acquired by Facebook 2013), and Kites, simultaneous translation services (acquired by Zoom 2021). Waibel is a member of the National Academy of Sciences of Germany, Fellow of the IEEE and of ISCA, and Research Fellow at Zoom. He holds BS/MS/PhD degrees from MIT and CMU.

Prof. Najim Dehak,
Johns Hopkins University, USA

Najim Dehak

An expert in machine learning and speech processing/speaker identification, Najim Dehak is internationally known as the lead developer of I-vector, a factor analysis-based speaker recognition technique. His research focuses on speech processing and modeling, audio segmentation, speaker, language, and emotion recognition. One of his interests has been building robust emotion detection systems that can be useful in several areas, including call centers, mental health, and social applications. He is also currently interested in working on topics related to human aging. In this topic, Dr. Dehak and his team are developing non-invasive, artificial intelligence-based tools to detect, assess, and monitor the functional and cognitive decline of elderly adults.